National Library
of Canada

Bibliothèque nationale
du Canada

Canadian Theses Service

Services des thèses canadiennes

Ottawa, Canada
K1A 0N4

CANADIAN THESES

THÈSES CANADIENNES

NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30.

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30.

THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED

LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE

NL-339 (r.86/06)

Canada

THE UNIVERSITY OF ALBERTA

Design and Evaluation of Practical Self-Tuning PID Controllers

by

©

Peter John Vermeer

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF Master of Science

IN

Process Control

Department of Chemical Engineering

EDMONTON, ALBERTA

Spring 1987

Permission has been granted to the National Library of Canada to microfilm this thesis and to lend or sell copies of the film.

The author (copyright owner) has reserved other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without his/her written permission.

L'autorisation a été accordée à la Bibliothèque nationale du Canada de microfilmer cette thèse et de prêter ou de vendre des exemplaires du film.

L'auteur (titulaire du droit d'auteur) se réserve les autres droits de publication; ni la thèse ni de longs extraits de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation écrite.

ISBN 0-315-37803-4

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR Peter John Vermeer

TITLE OF THESIS Design and Evaluation of Practical Self-Tuning PID Controllers

DEGREE FOR WHICH THESIS WAS PRESENTED Master of Science

YEAR THIS DEGREE GRANTED Spring 1987

Permission is hereby granted to THE UNIVERSITY OF ALBERTA LIBRARY to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

(SIGNED) ..P.J. Verme.......

PERMANENT ADDRESS:

295 Edgewood Street #41

Sarnia, Ontario

N7S 5A6

DATED ..March 20.....1987

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled Design and Evaluation of Practical Self-Tuning PID Controllers submitted by Peter John Vermeer in partial fulfilment of the requirements for the degree of Master of Science in Process Control.

....

Supervisor

....

....

Date..March 20, 1987..........

To my wife Grace and
my children, Michael and Maria

Abstract

Self-tuning PI(D) controllers have received considerable attention in recent years. This thesis analyzes three of these controllers which are based on a ARIMA model process representation: Song et al. (1986); Tuffs and Clarke (1985); Tjokro and Shah (1985). The first two controllers are simplified forms of the generalized minimum variance (GMV) controllers while the third is based on the pole placement technique. The stability and performance aspects of the algorithms are compared by means of simulation studies and root locus analysis. The comparison shows the pole placement based PI(D) controller to have superior stability and performance characteristics. In addition the Song et al. (1986) and the Tuffs and Clarke (1985) adaptive controllers are shown to be equivalent for the simplified explicit PI(D) forms.

An alternate design method is suggested for calculation of the controller gain which is directly related to the maximum overshoot of the closed-loop system when used in conjunction with the pole placement design principle. The method is based on the location of the closed-loop poles calculated from the estimated characteristic equation and exactly specifies the peak overshoot for low order processes. Simulation studies in the presence of process-model mismatch show the resultant self-tuning PI(D) controller to be a workable scheme.

Finally, process identification based on incremental variables is compared with that based on positional or absolute variables. Incremental identification is shown to be less sensitive to process bias and unmeasurable deterministic step disturbances than positional variables estimation, but is more sensitive to unmodelled dynamics, process nonlinearities, time delay mismatch and process noise.

Acknowledgements

The author wishes to express his sincere gratitude to Dr. A.J. Morris and Dr. S.L. Shah for their encouragement, help and supervision during the course of this research. Thanks are also due to Dr. D.G. Fisher for his direction and understanding in the absence of Dr. Morris and Dr. Shah.

I also owe much to conversations with my fellow graduate students in Process Control; in particular, Greg McDougall, Andre Vien, Jim Langman and Kirthi Walgama. Thanks are due to Paul Barrow, Bob Barton and Henry Sit of the DACS centre for their assistance in using the Department computer equipment and for many discussions which increased my knowledge of computer hardware and software

The author would like to thank the Department of Chemical Engineering and the Natural Sciences and Engineering Research Council of Canada for their financial support.

Finally, I would like to acknowledge the support and help of my wife Grace in the completion of this thesis. Her willingness to help and encourage are greatly appreciated.

Table of Contents

Chapter Page

Abstract v

Acknowledgements vii

List of Tables xii

List of Figures xiii

Nomenclature xviii

1. Introduction 1

1.1 Overview 1

1.2 Organization of Thesis 3

2. Proportional Plus Integral Plus Derivative Control ..6

2.1 Introduction 6

2.2 PID Control Law Derivation 7

2.3 Tuning of PID Controllers 15

3. Self-Tuning Control 18

3.1 Introduction 18

3.1.1 General Overview 18

3.1.2 Time Series Polynomial Models 20

3.2 The Self-Tuning Robust Controller 22

3.2.1 Introduction 22

3.2.2 Derivation of a Self-Tuning Robust Controller Based on the ARIMA Model 23

3.2.3 Simplification to a Self-Tuning PID Form 28

3.3 Integrating Self-Tuning Controller 31

3.3.1 Introduction 31

3.3.2 Derivation of the Integrating Form of the Self-Tuning Controller 31

3.3.3 Simplification to a Self-Tuning PID Form 33

3.4 Self-Tuning Pole Placement Controller 35

3.4.1 Introduction 36

3.4.2 Derivation of the Self-Tuning Pole Placement Controller 36

3.4.3 Simplification to a Self-Tuning PID Form 38

3.5 Comparison of Self-Tuning Control PID Algorithms 41

3.6 Alternate Method for Controller Gain Calculation 43

4. Additional Techniques for Enhanced Self-Tuning Control 47

4.1 Introduction 47

4.2 Choice of Sampling Period 47

4.3 Recursive Parameter Estimation 52

4.3.1 Parameter Estimation Methods 53

4.3.2 Discounting Old Data 56

4.3.3 Estimation Algorithm Conditioning 58

4.3.4 Nonzero Mean Data 59

4.3.5 Type of Identification 61

4.4 Model Reduction Technique 63

4.5 Filtering 68

4.6 Time Delay Compensation 71

4.7 Adaptive Feedforward Control 76

5. Linear SISO Process Models for Simulation Studies ..81

5.1 Introduction 81

5.2 Process Models 83

6. Stability and Performance - A Root Locus and Time-Domain Analysis 86

6.1 Introduction 86

6.2 The Root Locus Technique 88

6.3 Method of Comparison - A Typical Data Set 90

6.4 Comparison of GMV and PP Based PID Controllers 95

6.5 Comparison of GMV and PP Based PID Controllers 107

6.6 Summary 117

7. Performance of Self-Tuning PID Algorithm with Linear Systems: In the Presence of Process/Model Order Mismatch 124

7.1 Introduction 124

7.2 Self-Tuning PID Algorithm Implementation 125

7.3 Simulation Results for No Process/Model Mismatch 128

7.4 Simulation Results for Process/Model Mismatch 138

7.4.1 PI Control and Mismatch 138

7.4.2 PID Control and Mismatch 153

7.5 Summary 175

8. Performance of Self-Tuning PID Algorithm with Linear Systems: In the Presence of Delays and Disturbances 178

8.1 Introduction 178

8.2 Simulation Results for Processes with Time Delays 178

8.2.1 Controller Detuning 179

8.2.2 Time Delay Compensation 194

8.3 Simulation Results for Deterministic Disturbances 208

8.3.1 Process Bias 211

8.3.2 Load Disturbances 218

8.4 Simulation Results for Stochastic Disturbances......................................233

8.5 Summary 239

9. Performance of Self-Tuning PI Controller on a Simulated Nonlinear System 242

9.1 Introduction 242

9.2 Distillation Column Simulator 243

9.3 Overhead Composition Control 244

9.4 Bottoms Composition Control 255

9.5 Summary 264

10. Conclusions and Recommendations 267

10.1 Conclusions 267

10.2 Recommendations for Future Work 270

References 274

Appendix A 281

List of Tables

Table Page

3.1 Comparison of self-tuning PID algorithms 42

5.1 Linear SISO process models for simulation studies ...84

6.1 Comparison of GMV and PP based PI control laws 96

6.2 Comparison of GMV and PP based PID control laws 108

6.3 Comparison of PP and modified PP based PID control laws 120

7.1 Initialization parameters for self-tuning PI(D) controller 127

7.2 Performance of self-tuning PI controller in the presence of process/model order mismatch 140

7.3 Online tuning of M_p setpoint for self-tuning PI control 150

7.4 Performance of self-tuning PID controller in the presence of process/model order mismatch 154

7.5 Online tuning of M_p setpoint for self-tuning PID control 161

7.6 Effect of identification method on self-tuning PID control in the presence of process/model mismatch ..165

7.7 Effect of identification method on estimated model in the presence of process/model mismatch 173

8.1 Performance of self-tuning PID control in the presence of time delays 180

8.2 Performance of self-tuning PID control in the presence of time delay compensation (TDC) 198

8.3 Performance of self-tuning PI control for processes with nonzero mean levels 212

8.4 Regulatory performance of self-tuning PID control for unmeasurable load disturbances 223

9.1 Servo self-tuning PI control of overhead and bottoms composition of distillation column simulator (DCSP) 246

A.1 Input data file for distillation column simulator (DCSP) 282

List of Figures

Figure Page

2.10 Servo response of "setpoint-on-P&I&D" (——), "setpoint-on-P&I" (----), and "setpoint-on-I-only" (····) algorithms (K_c=4.7,T_i=7.06,T_d=1.62) 12

4.1 Equivalent block diagram of the Smith predictor 73

6.1 Root loci of Process Model 2 with a) GMVPID and b) PPPID controllers (P=M=C=2) 93

6.2 Servo response of Process Model 2 with GMVPID (----) and PPPID (——) controllers (P=M=C=2) 94

6.3 Root loci of Process Model 1 with a) GMVPI and b) PPPI controllers (P=M=C=1) 98

6.4 Servo response of Process Model 1 with GMVPI (----) and PPPI (——) controllers (P=M=C=1) 99

6.5 Root loci of Process Model 2 with a) GMVPI and b) PPPI controllers (P=M=2,C=1) 100

6.6 Root loci of Process Model 3 with a) GMVPI and b) PPPI controllers (P=3,M=2,C=1) 101

6.7 Servo response of Process Model 3 with GMVPI (----) and PPPI (——) controllers (P=3,M=2,C=1) 102

6.8 Root loci of Process Model 4 with a) GMVPI and b) PPPI controllers (P=3,M=2,C=1) 103

6.9 Root loci of Process Model 5 with a) GMVPI and b) PPPI controllers (P=M=2,C=1) 104

6.10 Root loci of Process Model 6 with a) GMVPI and b) PPPI controllers (P=3,M=2,C=1) 105

6.11 Root loci of Process Model 7 with a) GMVPI and b) PPPI controllers (P=M=2,C=1) 106

6.12 Root loci of Process Model 3 with a) GMVPID and b) PPPID controllers (P=3,M=C=2) 110

6.13 Servo response of Process Model 3 with GMVPID (----), PPPID (——) and modified PPPID (····) controllers (P=3,M=C=2) 111

6.14 Root loci of Process Model 4 with a) GMVPID and b) PPPID controllers (P=3,M=C=2) 112

6.15 Root loci of Process Model 5 with a) GMVPID and b) PPPID controllers (P=M=C=2) 113

6.16 Root loci of Process Model 6 with a) GMVPID and b) PPPID controllers (P=3,M=C=2) 114

6.17 Root loci of Process Model 7 with a) GMVPID and b) PPPID controllers (P=M=C=2) 115

6.18 Root loci of Process Model 3 with modified PPPID controller (P=3,M=C=2) 122

6.19 Root loci of Process Model 4 with modified PPPID controller (P=3,M=C=2) 122

6.20 Root loci of Process Model 6 with modified PPPID controller (P=3,M=C=2) 123

7.1 Servo self-tuning PI control of Process Model 1 (INC,P=M=C=1,$M_p$sp=15) 130

7.2 Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15) 131

7.3 Servo self-tuning PID control of Process Model 3 (INC,P=M=C=2,$M_p$sp=15) 134

7.4 Servo self-tuning PID control of Process Model 7 (INC,P=M=C=2,$M_p$sp=15) 135

7.5 Online Mp setpoint (15-30%) tuning of self-tuning PID controller for Process Model 2 137

7.6 Servo self-tuning PI control of Process Model 2 (INC,P=M=2,C=1,$M_p$sp=15) 141

7.7 Servo self-tuning PI control of Process Model 3 (INC,P=3,M=2,C=1,$M_p$sp=15) 142

7.8 Servo self-tuning PI control of Process Model 4 (INC,P=3,M=2,C=1,$M_p$sp=15) 143

7.9 Servo self-tuning PI control of Process Model 5 (INC,P=M=2,C=1,$M_p$sp=15) 145

7.10 Servo self-tuning PI control of Process Model 6 (INC,P=3,M=2,C=1,$M_p$sp=15) 146

7.11 Servo self-tuning PI control of Process Model 7 (INC,P=M=2,C=1,$M_p$sp=15) 147

7.12 Servo self-tuning PI control of Process Model 8 (INC,P=M=2,C=1,$M_p$sp=15) 148

7.13 Online M_p setpoint (15-30%) tuning of self-tuning PI controller for Process Model 3 151

7.14 Online M_p setpoint (15-30%) tuning of self-tuning PI controller for Process Model 4 152

7.15 Servo self-tuning PID control of Process Model 1 (INC,P=1,M=C=1,$M_p$sp=15) 155

7.16 Servo self-tuning PID control of Process Model 3 (INC,P=3,M=C=2,$M_p$sp=15) 156

7.17 Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=15) 157

7.18 Servo self-tuning PID control of Process Model 6 (INC,P=3,M=C=2,$M_p$sp=15) 159

7.19 Online M_p setpoint (15-30%) tuning of self-tuning PID controller for Process Model 3 162

7.20 Online M_p setpoint (15-30%) tuning of self-tuning PID controller for Process Model 4 163

7.21 Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=30) 166

7.22 Servo self-tuning PID control of Process Model 4 (POS,P=3,M=C=2,$M_p$sp=15) 168

7.23 Servo self-tuning PID control of Process Model 4 (POS,P=3,M=C=2,$M_p$sp=40) 170

7.24 Open-loop response of Process Model 3 (——) and estimated second order models from incremental (----) and positional (····) identification 174

7.25 Open-loop response of Process Model 4 (——) and estimated second order models from incremental (----) and positional (····) identification 174

8.1 Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15-30,da=de=1) 181

8.2 Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15,da=1,de=0) 183

8.3 Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=20,da=0,de=1) 185

8.4 Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=40,da=de=4) 187

8.5 Root loci of Process Model 2 with selt-tuning PID controller for time delay = 4 188

8.6 Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=15,da=de=1,T_s=1) 190

8.7 Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=15,da=1,de=0,T_s=1) 191

8.8 Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=15,da=1,de=0-1,T_s=1) 192

8.9 Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=40,da=de=4,T_s=1) 195

8.10 Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=40,da=de=4,T_s=2) 196

8.11 Servo self-tuning PID control of Process Model 2 with time delay compensator (INC,P=M=C=2,$M_p$sp=15,da=de=4,TDC) 199

8.12 Servo self-tuning PID control of Process Model 2 with time delay compensator (INC,P=M=C=2,$M_p$sp=15,da=2,4,de=2-5,TDC) 202

8.13 Servo self-tuning PID control of Process Model 4 with time delay compensator (INC,P=3,M=C=2,$M_p$sp=30,da=de=4,TDC,T_s=1) 204

8.14 Servo self-tuning PID control of Process Model 4 with time delay compensator (INC,P=3,M=C=2,$M_p$sp=30,da=de=4,TDC,T_s=2) 206

8.15 Servo self-tuning PID control of Process Model 4 with time delay compensator (INC,P=3,M=C=2,$M_p$sp=30,da=2,4,de=2-5,TDC,T_s=2) 209

8.16 Servo self-tuning PI control of Process Model 2 (POS,P=M=2,C=1,$M_p$sp=15,bias=0.0) 214

8.17 Servo self-tuning PI control of Process Model 2 (INC,P=M=2,C=1,$M_p$sp=15,bias=4.36) 215

8.18 Servo self-tuning PI control of Process Model 2 (POS,P=M=2,C=1,$M_p$sp=15,bias=4.36) 216

8.19 Servo self-tuning PI control of Process Model 4 (POS,P=3,M=2,C=1,$M_p$sp=15,bias=0.0) 219

8.20 Servo self-tuning PI control of Process Model 4 (POS,P=3,M=2,C=1,$M_p$sp=15,bias=2.76) 220

8.21 Regulatory self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15,load@50,150) 224

8.22 Regulatory self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=30,load@50,150) 226

8.23 Regulatory self-tuning PID control of Process Model 2 (POS,P=M=C=2,$M_p$sp=15,load@50,150) 229

8.24 Regulatory self-tuning PID control of Process Model 4 (POS,P=3,M=C=2,$M_p$sp=30,load@50,150) 231

8.25 Servo self-tuning PI control of Process Model 1 (INC,P=M=C=1,$M_p$sp=15,noise) 235

8.26 Servo self-tuning PI control of Process Model 4 (INC,P=3,M=2,C=1,$M_p$sp=15,noise) 237

9.1 Servo self-tuning PI control of OVHD composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,no noise) 247

9.2 Servo self-tuning PI control of OVHD composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,noise) 249

9.3 Servo self-tuning PI control of OVHD composition (INC,P=M=C=1,$M_p$sp=15,T_s=1,no noise) 251

9.4 Servo self-tuning PI control of OVHD composition (INC,P=M=C=1,$M_p$sp=15,T_s=1,noise) 253

9.5 Regulatory PI control of OVHD composition (K_c=0.793,T_i=2.63,T_s=3,noise) 256

9.6 Servo self-tuning PI control of BTMS composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,da=de=1,no noise) 258

9.7 Servo self-tuning PI control of BTMS composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,da=de=1,noise) 260

9.8 Regulatory PI control of BTMS composition (K_c=0.232,T_i=8.60,T_s=3,noise) 263

Nomenclature

$A(z^{-1})$	Polynomial in the backward shift operator
$B(z^{-1})$	Polynomial in the backward shift operator
d	Process bias or process time delay excluding sampling delay ($d=k-1$)
$D(z^{-1})$	Polynomial in the backward shift operator
$D_1(z^{-1})$	Error feedback controller numerator polynomial
$D_2(z^{-1})$	Output feedback controller numerator polynomial
$E(z^{-1})$	Polynomial in the backward shift operator
$F(z^{-1})$	Polynomial in the backward shift operator
$G(z^{-1})$	Polynomial in the backward shift operator
$H(z^{-1})$	Polynomial in the backward shift operator
k	Process time delay including sampling delay ($k\geq1$)
$K(t)$	Gain vector for recursive least squares algorithm
K_c	Controller gain
K_p	Process gain
K_1	Constant defined for pole placement PID algorithm
K_2	Constant defined for pole placement PID algorithm
$L(z^{-1})$	Polynomial in the backward shift operator
M_p	Maximum overshoot
$N(z^{-1})$	Polynomial in the backward shift operator
$P(t)$	Covariance matrix of parameter estimates
$P(z^{-1})$	Rational weighting transfer function in the backward shift operator
r	Radius for polar coordinates

$Q(z^{-1})$	Rational weighting transfer function in the backward shift operator
$S(z^{-1})$	Polynomial in the backward shift operator
t_r	Rise time
t_s	Settling time
T_d	Derivative time
T_i	Integral time
T_s	Sampling interval
$u(t)$	Process input signal
$v(t)$	Measurable load disturbance signal
$w(t)$	Unmeasurable deterministic disturbance signal
$W(z^{-1})$	Polynomial in the backward shift operator
$x(t)$	Stochastic disturbance signal
$X(z^{-1})$	Polynomial in the backward shift operator
$y_r(t)$	Process reference signal
α	Constant defined for alternate K_c calculation method or first order digital filter constant
Δ	Differencing operator $1 - z^{-1}$
$\epsilon(t)$	Setpoint error signal
ζ	Damping ratio
$\eta(t)$	Auxiliary stabilizing signal for SFC
θ	Angle for polar coordinate or process time delay in the continuous-time domains
$\theta(t)$	Process parameter vector
λ	Intermediate result for pole placement PID algorithm
$\lambda(t)$	Variable forgetting factor
$\xi(t)$	Independent random variable sequence

$\tau(t)$	Process time constant
$\phi(t)$	Input/output regressor vector
ω_n	Natural frequency

Figure Titles

bias	Process bias or offset
C	Order of the control law : 1=PI, 2=PID
da	Actual time delay excluding unit sample delay (i.e. d = k - 1)
de	Estimated time delay excluding unit sample delay (i.e. d = k - 1)
GMVPI(D)	Generalized minimum variance based PI(D) controller
INC	Incremental variable identification
M	Order of the estimated process model
M_pSP	Maximum overshoot setpoint
P	Order of linear process transfer function model
POS	Positional or absolute variable identification
PPPI(D)	Pole placement based PI(D) controller
TDC	Time delay compensation
Ytdc	Predicted undelayed output from time delay compensator

1. Introduction

1.1 Overview

The initial intent of this thesis work was to investigate the performance of a single-input, single-output (SISO) self-tuning PI(D) controller in a multivariable environment. Interaction of the multi-output, multi-output (MIMO) was to be decoupled with an adaptive feedforward compensator. Time delays were to be dealt with by means of a time delay compensator (TDC). This multiloop strategy was to be implemented on a microcomputer. The process input-output subsystem was to be an industrial controller with a communication link to the microcomputer.

The work progressed on this initial research proposal. A number of multivariable systems were identified as potential simulation processes. Static and dynamic decouplers were designed to reduce the interaction of these multivariable systems by a direct Nyquist array frequency analysis. Familiarization with the microcomputer system, industrial controller and the communication protocol was completed. A driver program in PASCAL and Modula-2 to communicate between the personal computer and the industrial controller was written. A literature review of published self-tuning PI(D) algorithms suggested a control algorithm by Song (1983) be used in this study. Simulation software including a MIMO continuous-time linear plant simulator was then written to test the algorithm in a SISO and MIMO environment. The simulator allowed underdamped and overdamped process of up to third order with numerator dynamics and time delays. Options included load disturbances, process and measurement noise.

However, it soon became apparent from simulation studies that this self-tuning PI(D) controller calculated unstable controller gains for some processes and sampling rate choices. It now became necessary to determine the reason behind this problem. The resulting investigation revealed that a great deal of analysis work should be done on adaptive controllers to understand the "how" and "why" of their closed-loop performance before they are applied to multivariable systems.

This need for analysis of self-tuning PI(D) controllers shifted the course of the research work midstream. It now became important to analyze and compare a number of self-tuning PI(D) algorithms to determine which gives better performance and why. This evaluation proceeds along the lines of root locus analysis and time-domain simulations. The results of the evaluation are used to suggest modifications to improve the stability and performance characteristics of the self-tuning PI(D) algorithms. This then became the focus of the thesis work. Only after this analysis is complete is it appropriate to implement a self-tuning PID controller in a multiloop configuration for multivariable systems.

1.2 Organization of Thesis

The classical contiunous proportional plus integral plus derivative (PID) control algorithm is presented in Chapter 2. The positional and velocity forms of the discrete PID algorithm are derived. The effect of removinq the setpoint from the proportional and derivative terms of the controller are considered. The PID control algorithm is also presented as a pole-zero controller to assist in later root locus analysis. The chapter is completed by investigating tuning procedures for this algorithm. A brief overview of both classical tuning and auto-tuning techniques is presented.

Chapter 3 investigates self-tuning control as it relates to the PID algorithm form. A brief literature review is presented to set the context for the self-tuning controllers. Three adaptive controllers are derived and simplified into a self-tuning PI(D) form. Two of the controllers are based on the generalized minimum variance (GMV) principle; the third on the pole placement principle. The three PID forms are compared and an alternate method for controller gain calculation is also presented in the chapter.

Related topics for the implementation of self-tuning control are discussed in Chapter 4. The importance of the sampling rate choice is noted as well as guidelines for its selection. Various least squares estimation methods are presented with implementation issues such as details of the UDU^T factorization method, types of variables used in the input/output regressor, conditioning problems and filtering. The chapter is completed by considering a model reduction technique, a time delay compensation method and adaptive feedforward control.

In Chapter 5 eight linear single-input, single-output linear processes are outlined . The models are presented in transfer function form in terms of the complex Laplace and z operator. The choice of sampling time for each process is given.

A stability and performance analysis of the self-tuning PI(D) algorithm is done in Chapter 6 for the eight linear process models. The methods used for comparing the algorithms are the root locus technique and time-domain simulations.

Chapters 7 and 8 presents simulation results for the preferred self-tuning PI(D) controller on the linear systems. The effects of mismatch, time delays, process bias, load disturbances and noise are considered. In addition results are presented to compare incremental and positional variable identification.

The evaluation of the preferred self-tuning PI(D) algorithm is continued in Chapter 9 on a distillation process. Details of a nonlinear distillation column simulation package are presented along with results for SISO servo and regulatory control of the overhead and bottoms composition of the column model.

Chapter 10 summarizes the results of the entire thesis. Conclusions are drawn and recommendations for the direction of future work are outlined.

2. Proportional Plus Integral Plus Derivative Control

2.1 Introduction

Proportional-Integral-Derivative (PID) controllers, also known as three-term controllers, have been used widely in the process industries for over four decades. Originally, they were implemented using analog hardware but more recently many of these controllers have been implemented in digital form. By and large the advances in control theory and techniques over the past 20 years or so have not replaced or displaced the PID control algorithm in the process industries but have rather served to modify or enhance its basic characteristics. There are many reasons for this, a number of which have been summarized by Clarke and Gawthrop (1981) :

1. The PID control structure is a robust structure which is remarkably effective in regulating a wide range of processes. Low, middle and high frequency behavior are adjusted by the integral time, the controller gain and the derivative time respectively.
2. Many advanced control techniques are model-based. Equipment and processes in the process industries are generally less well understood and therefore more difficult to model than well-defined aerospace, electrical or mechanical control problems.
3. Nonlinearities, time delays and time-varying parameters tend to characterize "difficult" control problems in

the process industries.

4. The *a priori* economic benefits of modern control is often difficult to establish. It is therefore often difficult to justify the effort required to implement modern control techniques.
5. Many of the modern control techniques are unproven and may not be robust enough to handle industrial processes.

It is, however, quite true that PID controllers have dominated the process industries and will probably continue to dominate them. The purpose of this chapter is to provide a brief review of the mathematical representations of the various forms of the PID algorithm in both the continuous and discrete domains. This will be followed by a literature review of the various tuning methods available for PID controllers.

2.2 PID Control Law Derivation

A standard "textbook," continuous-time PID controller is written in the form:

$$u(t) = K_c[\epsilon(t) + \frac{1}{T_i}\int \epsilon(t)dt + T_d\frac{d\epsilon}{dt}] + u_s \qquad (2.1)$$

where K_c is the controller gain, T_i the integral or reset time and T_d the derivative or rate time. The controller output and the error signal are $u(t)$ and $\epsilon(t)$ respectively. The error signal is defined as

$$\epsilon(t) = y_r(t) - y(t) \tag{2.2}$$

where $y_r(t)$ and $y(t)$ are the process reference signal and output repectively. The controller bias, u_s, represents the mean output of the controller when the error is zero. Equation 2.1 can be rewritten in Laplace form (assume $u_s=0$) as:

$$U(s) = K_c[1 + \frac{1}{T_i s} + T_d s]\, E(s) \tag{2.3}$$

The derivation of a discrete PID approximation of Equation 2.1 and 2.3 can proceed along various lines (Isermann, 1981; Åström and Wittenmark ,1984). If Equation 2.1 is the starting point, a discrete PID algorithm can be deduced by approximating the derivative term by a first order backward difference and the continuous integration by either a rectangular or trapezoidal integration. If rectangular integration is chosen, the discrete PID algorithm has the following form:

$$u(t) = K_c[\epsilon(t) + \frac{T_s}{T_i}\sum_{0}^{t}\epsilon(i) + \frac{T_d}{T_s}(\epsilon(t)-\epsilon(t-1))] \tag{2.4}$$

A discrete PID algorithm can be derived from Equation 2.3 by straightforward sampling or by taking Euler or backward difference approximations to the Laplace complex variable. If the backward difference method is used, Equation 2.3 takes the form:

$$U(z) = K_c[1 + \frac{T_s z}{T_i(z-1)} + \frac{T_d(z-1)}{T_s z}] E(z) \qquad (2.5)$$

where z is the complex z-plane operator.

The discrete forms of Equations 2.4 and 2.5 are called the positional forms of the PID controller because the total output of the controller is calculated. A velocity or incremental form can be derived if a change in control signal is defined as:

$$\Delta u(t) = u(t) - u(t-1) \qquad (2.6)$$

The velocity form of Equation 2.4 then becomes

$$\Delta u(t) = K_c[(1+\frac{T_s}{T_i}+\frac{T_d}{T_s})-(1+2\frac{T_d}{T_s})z^{-1}+\frac{T_d}{T_s}z^{-2}]\epsilon(t) \qquad (2.7)$$

or

$$\Delta u(t) = K_c[\epsilon(t) - \epsilon(t-1) + \frac{T_s}{T_i}\epsilon(t) + \frac{T_d}{T_s}(\epsilon(t) - 2\epsilon(t-1) + \epsilon(t-2))] \qquad (2.8)$$

where z^{-1} is the backward shift operator. This form affords certain operational advantages over the positional form. It is a recursive algorithm more suitable to computer implementation. It does not require initialization for bumpless transfer from manual to automatic control. It has inherent antireset windup protection when the output is limited.

The structure of the "textbook" form of the PID controller (Equation 2.1) can be modified in several ways (Åström and Wittenmark, 1984). The structure of the algorithm discussed so far has the error signal driving the proportional, integral and derivative terms. This will be called the "setpoint-on-P&I&D" structure. A common modification to the basic algorithm is to remove the setpoint from the derivative term thus avoiding the well known "derivative kick" phenomenon during sudden setpoint changes in the reference signal. This is the structure most often seen in the literature. It will be referred to as the "setpoint-on-P&I" structure and is of the form:

$$\Delta u(t) = K_c[\epsilon(t) - \epsilon(t-1) + \frac{T_s}{T_i}\epsilon(t) + \frac{T_d}{T_s}(-y(t) + 2y(t-1) - y(t-2))] \qquad (2.9)$$

An alternative method to avoid "derivative kick" is to shape setpoint changes to follow a ramp or first order model.

A third structure, less commonly seen, is to remove the setpoint from the proportional term in addition to the derivative term. This is referred to as the "setpoint-on-I-only" structure and is of the form:

$$\Delta u(t) = K_c[-y(t) + y(t-1) + \frac{T_s}{T_i}\epsilon(t) + \frac{T_d}{T_s}(-y(t) + 2y(t-1) - y(t-2))] \qquad (2.10)$$

Other forms of PID controller structure are summarized in

Wittenmark (1979).

The structure of the PID control algorithm affects its closed-loop performance. The servo response of the "setpoint-on-P&I&D", "setpoint-on-P&I" and "setpoint-on-I-only" forms is shown in Figure 2.1 for a second order overdamped process with $K_c=4.7$, $T_i=7.06$ and $T_d=1.62$. Removing the setpoint from the proportional and derivative terms slows down the servo response. However, the regulatory performance of each structure is identical. For a unit step load disturbance the regulatory response is similiar in form to the "setpoint-on-P&I&D" servo response.

An explanation for the servo and regulatory response characteristics of the three PID control algorithm structures is easily seem by considering the PID control law to be represented by:

$$\Delta u(t) = D_1(z^{-1})\epsilon(t) - D_2(z^{-1})y(t) \qquad (2.11)$$

where $D_1(z^{-1})$ corresponds to the numerator polynomial of the error feedback controller, $D_2(z^{-1})$ corresponds to the numerator polynomial of the output feedback controller and z^{-1} is the backward shift operator. Consider also a deterministic process represented by:

$$A(z^{-1})y(t) = B(z^{-1})u(t) - L(z^{-1})v(t) \qquad (2.12)$$

where $A(z^{-1})$, $B(z^{-1})$ and $L(z^{-1})$ are polynomials in the backward shift operator and $v(t)$ is a process load

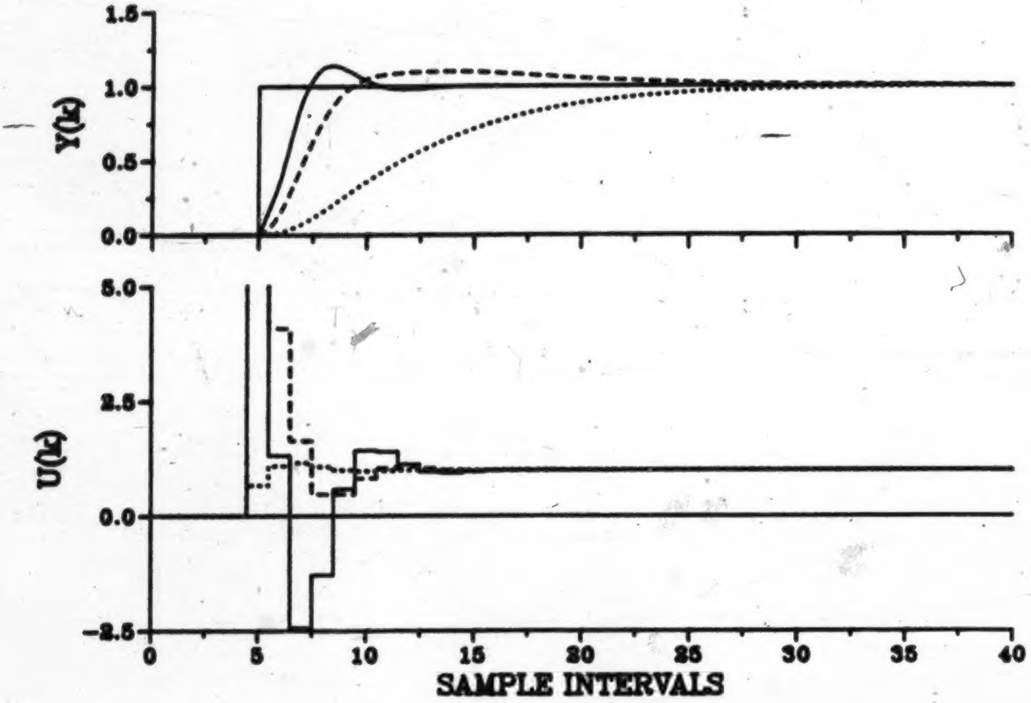

Figure 2.1 Servo response of "setpoint-on-P&I&D" (——), "setpoint-on-P&I" (----), and "setpoint-on-I-only" (····) algorithms (K_c=4.7, T_i=7.06, T_d=1.62)

disturbance. The closed-loop system transfer function relating y(t) and $y_r(t)$ is:

$$y(t) = \frac{B(z^{-1})D_1(z^{-1})}{\Delta A(z^{-1}) + B(z^{-1})[D_1(z^{-1})+D_2(z^{-1})]} y_r(t) \qquad (2.13)$$

The closed-loop load transfer function relating y(t) and v(t) is:

$$y(t) = \frac{\Delta L(z^{-1})}{\Delta A(z^{-1}) + B(z^{-1})[D_1(z^{-1})+D_2(z^{-1})]} v(t) \qquad (2.14)$$

It can be easily shown that the denominator of the closed-loop characteristic polynomial is invariant or independent of the partitioning of the of the numerator polynomial of the controller into terms $D_1(z^{-1})$ and $D_2(z^{-1})$ since it is the sum of $D_1(z^{-1})$ and $D_2(z^{-1})$ which appears in the characteristic polynomial. Therefore, the stability boundary for the three PID algorithm structures is the same. The numerator of the closed-loop system transfer function determines the zeroes of the closed-loop system, and hence influences the transient response characteristics. These zeroes vary depending on the partitioning of the controller. For the "setpoint-on-P&I" structure $D_1(z^{-1})$ is:

$$D_1(z^{-1}) = K_c[(1 + T_s/T_i) - z^{-1}] \qquad (2.15)$$

and for the "setpoint-on-I-only" structure $D_1(z^{-1})$ is:

$$D_1(z^{-1}) = K_c T_s / T_i \tag{2.16}$$

This accounts for the different servo responses of the three controller structures. The numerator polynomial of the closed-loop load transfer function is also invariant of the controller numerator polynomial and accounts for the identical regulatory response of the three controller structures.

The PID controller can also be viewed from a pole-zero perspective. If the discrete velocity PID algorithm in Equation 2.7 is examined from this standpoint, its transfer function takes the form:

$$G_c(z) = \frac{K_c\left[\left(1+\frac{T_s}{T_i}+\frac{T_d}{T_s}\right)z^2 - \left(1+2\frac{T_d}{T_s}\right)z + \frac{T_d}{T_s}\right]}{z\,(z - 1)} \tag{2.17}$$

The controller poles are fixed in the z-plane unit circle at $z=0$ and $z=1$. The controller zeroes are determined by the integral and derivative times and sampling time. If $T_i>4T_d$, the two controller zeroes will be real and distinct. If $T_i=4T_d$, the two controller zeroes will be real and equal and if $T_i<4T_d$, the two controller zeroes will be a complex conjugate pair. It will be noted that the PID algorithm in Equation 2.7 introduces two zeroes into the closed-loop transfer function. Wittenmark (1979) shows some PID structures which introduce one, or even no, additional zeroes into the closed-loop transfer function.

The analysis of this work will focus on the discrete velocity "setpoint-on-P&I&D" PID structure shown in Equations 2.7, 2.8 and 2.17.

2.3 Tuning of PID Controllers

The PID algorithm is a robust, simple algorithm which can handle most of the control loops in the process industries but must be tuned properly to obtain "reasonable" closed-loop control. Over the past 40 years many tuning methods have been proposed. However, a large percentage of the process control loops in industry are still poorly tuned or even set on manual (Andreiev, 1981).

Ziegler and Nichols (1942) proposed the first two manual methods for tuning PID control algorithms. The transient-response method calculates the steepest slope, R, and delay time, L, from a unit-step response of the open-loop system (i.e. process reaction curve). The controller settings are then calculated from R and L. The ultimate-sensitivity method involved closed-loop control of the plant under proportional control only. The gain is increased until sustained oscillations are obtained. Controller settings are then calculated from the ultimate gain and period of oscillation.

Cohen and Coon (1953) suggested controller parameters based on a first order plus dead time (FOPDT) model of the process reaction curve. More recently other model-based methods have been proposed by Kuo, Corripio and Smith (1973) and MacGregor, Wright and Hong (1975). Smith and Corripio

(1985) summarize tuning formulae based on the integral of error squared (ISE), integral of absolute error (IAE), and integral of time absolute error (ITAE) criteria.

Most recently several new auto-tuning methods have been proposed. Åström and Hägglund (1984) have automated the Ziegler-Nichols method by proposing a different technique to determine the critical gain and critical period of the closed-loop system. It involves closed-loop relay control. The critical gain and period are determined from the input signal excited by the relay amplitude and the amplitude of oscillation of the system output. Gain and phase margin methods can also be used. Hägglund and Åström (1985) have extended this method by determining two points on the Nyquist curve and using dominant pole design to calculate the PID settings. NAF Controls have introduced a commercial controller based on this type of method (Åström, 1985).

Another auto-tuning method has been proposed by Nishikawa et al. (1984). It involves sampling a process response to a test input signal under open-loop or closed-loop conditions, processing the data to estimate characteristic values of the process and calculating the PID control settings by optimization of a weighted ISE function. The Foxboro EXACT controller is a more recent entry into the 'auto-tuning' controller market. PID settings are based on pattern recognition of the process.

PID tuning using model estimation have been by far the most common methods introduced in recent years. A common approach taken is to derive a complex controller based on some commonly accepted design method (i.e. generalized minimum variance, pole placement, linear quadratic gaussian control, optimization methods, phase and gain margin specification). A time series model of the process is recursively estimated on-line. If sufficient simplifying conditions are imposed, a 3-term PID controller structure can be obtained. The estimated parameters can then be used to calculate the controller settings. These techniques can be found in Kohahl and Isermann (1985), Tuffs and Clarke (1985), Song et al. (1983, 1984, 1986) and Tjokro and Shah (1985). Commercial controllers on the market that fit into this category are the Leeds and Northrup Electromax V, the Turnbull Control System TCS 6355, and the ASEA Novatune. Recently Dumont et al. (1985) and Zervos et al. (1985) have proposed a novel method for automatic tuning of PID controllers. Here the closed-loop system is represented by a series of Laguerre functions and the PID settings are found by optimization of an ISE criteria.

3. Self-Tuning Control

3.1 Introduction

This chapter sets the overall context for adaptive control and derives three self-tuning PID controllers which will be evaluated and compared. The remainder of Section 3.1 presents a brief overview of adaptive control and the linear models used to describe the process for adaptive control. Section 3.2 derives a PID controller from the self-tuning robust (SRC) controller of Song et al. (1986). The integrating form of the Clarke and Gawthrop self-controller (STC) (Tuffs and Clarke, 1985) is developed in Section 3.3. This controller is then simplified into its PID form. A self-tuning PID controller based on a pole placement design is derived in Section 3.4. The explicit forms of these adaptive PID algorithms are compared in Section 3.5. Section 3.6 proposes a new method to calculate the controller gain.

3.1.1 General Overview

The concept of adaptive control goes back thirty years to the late 1950's when there was a great interest in designing autopilots for high performance rockets and aircraft capable of handling a wide range of flight conditions. Some early papers in the field were those of Whitaker et al. (1958), Kalman (1958) and Gregory (1959). These efforts were largely unsuccessful due to insufficient theory and poor hardware and therefore interest in the area waned.

There was a renewed interest in self-tuning control systems in the 1970's. The 1960's had witnessed the theoretical developments of modern control concepts, especially those of stability theory, while the early 1970's saw the widespread availability of inexpensive digital computer hardware. Major steps forward were made by the self-tuning controller systems of Peterka (1970), Åström and Wittenmark (1973) and Clarke and Gawthrop (1975).

Since this time adaptive control has become a very active field of research. For a general overview of the area the reader is referred to survey articles by Seborg et al. (1986) and Åström (1983).

The adaptive control strategies which appear in the literature may be classified by considering the design method used for their derivation. Three main categories become apparent and are listed below:

1. Controllers based on the minimization of a quadratic cost function (e.g. self-tuning regulators and controllers). Many simple (predictor based) adaptive techniques have been shown to be globally stable at least in the deterministic case.
2. Controllers derived from a stability point of view (e.g. model reference adaptive control)
3. Controllers based on pole-zero assignment techniques (e.g. Åström and Wittenmark, 1980).

A further classification of adaptive controller is possible. This classification is not based on the theory or technique from which the controller is derived but how the algorithm is implemented from the viewpoint of parameter estimation. The two general classes are: explicit or indirect methods, and implicit or direct methods. In the explicit approach the process parameters are estimated and used in the controller design calculations. It is called indirect because the controller parameter are derived from the process parameters. In the implicit or direct approach the original process model is manipulated into a predictive model where the controller parameters are estimated directly from the input-output data.

The adaptive controllers developed in this chapter can be conveniently characterized by the classification detailed above. The self-tuning feedback controller (Song et al., 1986) and the integrating self-tuning controller (Tuffs and Clarke, 1985) are implicit controllers derived form the minimization of a quadratic cost function. The adaptive controller of Tjokro and Shah (1985) is an explicit controller based on a pole placement design.

3.1.2 Time Series Polynomial Models

The development of a self-tuning control policy generally starts with the assumption that a plant can be represented, under sampled-data control, by a locally linearized model of the form:

$$A(z^{-1})y(t) = z^{-k}B(z^{-1})u(t) + x(t) \qquad (3.1)$$

in which $A(z^{-1})$ and $B(z^{-1})$ are polynomials in the backward shift operator z^{-1} represented by:

$$\begin{aligned} A(z^{-1}) &= 1 + a_1z^{-1} + a_2z^{-2} + \dots + a_{n_a}z^{-n_a} \\ B(z^{-1}) &= b_0 + b_1z^{-1} + b_2z^{-2} + \dots + b_{n_b}z^{-n_b} \end{aligned} \qquad (3.2)$$

and k is the system time delay specified as an integer number of sample intervals ($k \geq 1$). The plant input and output are u(t) and y(t) respectively. The signal x(t) models the disturbances acting on the process and generally takes the form $C(z^{-1})\xi(t) + d$ so that Equation 3.1 becomes:

$$A(z^{-1})y(t) = z^{-k}B(z^{-1})u(t) + C(z^{-1})\xi(t) + d \qquad (3.3)$$

where $\xi(t)$ is a sequence of independent random variables, d is the process bias and $C(z^{-1})$ is a polynomial represented by:

$$C(z^{-1}) = 1 + c_1z^{-1} + c_2z^{-2} + \dots + c_{n_c}z^{-n_c} \qquad (3.4)$$

The disturbance process is also generally assumed to be stationary with zero mean. The complete model is known as a 'Autoregressive Moving Average' or ARMA model.

A common problem which must be faced when deriving self-tuning controllers based on the ARMA model is the problem of offset and nonstationary data. An approach often used in the identification literature to handle these situations is to postulate a disturbance process which has stationary increments (Kawashima, 1981; Yaglom, 1973):

$$x(t) = C(z^{-1})\xi(t)/\Delta + d \tag{3.5}$$

where Δ is the differencing operator $(1-z^{-1})$. Combining Equations 3.1 and 3.3 results in the ARIMA (Autoregressive Integrating Moving Average) system representation:

$$A(z^{-1})y(t) = z^{-k}B(z^{-1})u(t) + C(z^{-1})\xi(t)/\Delta + d \tag{3.6}$$

By assuming various forms for $C(z^{-1})$ and $\xi(t)$, one can interpret $x(t)$ as either stochastic (Brownian motion) or representing random steps. Another feature of the ARIMA model is the fact that integrating control arises naturally.

Most recently Tuffs and Clarke (1985) have suggested that the ARIMA model be used in the derivation of self-tuning controllers. Other workers (Belanger, 1983; Fessel and Karny, 1979; Gawthrop, 1982a; Gawthrop, 1982b) have also suggested its use. The controllers derived in this thesis are based upon the ARIMA model.

3.2 The Self-Tuning Robust Controller

3.2.1 Introduction

A self-tuning robust controller (SRC) has been described in Song et al. (1983, 1984, 1986). The controller is based on the internal model principle of Francis and Wonham (1975, 1976) which states that a compensator can achieve stability plus steady state regulation and/or tracking despite certain finite perturbations in the system parameters only if the controller utilizes feedback of the regulated variable, and incorporates in the feedback path a suitability reduplicated model of the dynamic structure of the exogenous signals which it is required to process.

Davison (1976) demonstrated that such a controller can be realized using separate "servo" and "stabilizing" compensators. The specific SRC proposed by Song et al. generalizes and extends the robust control strategy of Silveira and Doraiswami (1984). In summary it is an error-driven, "robust" structure controller which incorporates an internal model of the reference and disturbance signals.

Song's SRC was derived using a ARMA representation of the process. The purpose of this section is to rederive the SRC using a ARIMA process model. The derivation will be in summary form. If more details are required, the reader is referred to the original derivation in Song et al. (1983, 1986)

3.2.2 Derivation of a Self-Tuning Robust Controller Based on the ARIMA Model

It is assumed that the process to be controlled can be described by a discrete equation of the form:

$$A_m(z^{-1})y(t) = z^{-k}B_m(z^{-1})u(t) + H_m(z^{-1})w(t) + x(t) \quad (3.7)$$

in which $A_m(z^{-1})$, $B_m(z^{-1})$ and $H_m(z^{-1})$ are polynomials in the backward shift operator z^{-1}. $A_m(z^{-1})$ is monic and the system delay is denoted by k. The process input and output are u(t) and y(t) respectively. The unmeasured deterministic disturbance sequence is given by w(t). The residual, x(t), is defined as the difference between the actual output and the deterministic model and can be any bounded deterministic or stochastic signal.

If it is assumed that x(t) is of the form:

$$x(t) = C_m(z^{-1})\xi(t)/\Delta \tag{3.8}$$

where $C_m(z^{-1})$ is a monic polynomial, Δ is the differencing operator and $\xi(t)$ is a sequence of independent random variables, the ARIMA process representation is obtained:

$$A_m y(t) = z^{-k} B_m u(t) + H_m w(t) + C_m \xi(t)/\Delta \tag{3.9}$$

The SRC derivation assumes that $y_r(t)$ and w(t) asympotically satisfy the following relationship (Davison, 1976; Silveira and Doraiswami, 1984):

$$\begin{aligned} D(z^{-1})y_r(t) &= 0 \\ D(z^{-1})w(t) &= 0 \end{aligned} \tag{3.10}$$

where $y_r(t)$ is the reference signal and $D(z^{-1})$ is a monic polynomial with known coefficients which characterizes the dynamics of the reference signal and the unmeasured deterministic disturbances.

The control error $\epsilon(t)$ is defined as:

$$\epsilon(t) = y_r(t) - y(t) \qquad (3.11)$$

If Equation 3.11 is multiplied by $D(z^{-1})$ and a substitution is made for $y(t)$ from Equation 3.9, it can be shown that:

$$-A_m D\epsilon(t) = z^{-k}B_m Du(t) + C_m D\xi(t)/\Delta \qquad (3.12)$$

In addition to the process description considered in Equation 3.12, the following structure is required for the adaptive controller:

$$u(t) = \frac{P(z^{-1})}{D(z^{-1})}\epsilon(t) + \frac{1}{D(z^{-1})}\eta(t) \qquad (3.13)$$

The first term of the RHS corresponds to the servo-compensator output and the second term to the stabilizing compensator output. $P(z^{-1})$ is a rational weighting transfer function chosen by the user and $\eta(t)$ is an auxiliary signal to be defined later.

Substitution of Equation 3.13 into 3.12 yields:

$$-[A_m D + z^{-k}B_m P]\epsilon(t) = z^{-k}B_m\eta(t) + C_m D\xi(t)/\Delta \qquad (3.14)$$

To simplify further analysis Equation 3.14 can be written more compactly by defining the following new polynomials:

$$\begin{aligned} A(z^{-1}) &= -[A_m(z^{-1})D(z^{-1}) + z^{-k}B_m(z^{-1})P(z^{-1})] \\ B(z^{-1}) &= B_m(z^{-1}) \qquad (3.15) \\ C(z^{-1}) &= C_m(z^{-1})D(z^{-1})/\Delta \end{aligned}$$

where n_a, n_b and n_c are the orders of the $A(z^{-1})$, $B(z^{-1})$ and $C(z^{-1})$ polynomials respectively. Equation 3.14 can now be rewritten as:

$$\epsilon(t) = \frac{B}{A}\eta(t-k) + \frac{C}{A}\xi(t) \qquad (3.16)$$

The auxiliary signal is determined by minimizing the following performance index J:

$$J = E\{[P(z^{-1})\epsilon(t+k)]^2 + [Q'(z^{-1})D(z^{-1})u(t)]^2\} \qquad (3.17)$$

where

$$P(z^{-1}) = [P_n(z^{-1})/P_d(z^{-1})]$$

$$Q'(z^{-1}) = [Q'_n(z^{-1})/Q'_d(z^{-1})]$$

are rational transfer functions weightings on system output error and control effort respectively. To minimize this performance index, $\epsilon(t+k)$ must be expressed in terms of known values at time t. Equation 3.16 can be rewritten as a weighted, predicted control error:

$$P\epsilon(t+k) = \frac{PB}{A}\eta(t) + \frac{PC}{A}\xi(t+k) \qquad (3.18)$$

The well known Diophantine identity can be used to separate $\xi(t+k)$ into present and future parts:

$$\frac{PC}{A} = E + \frac{z^{-k}F}{P_dA} \qquad (3.19)$$

where

$$E(z^{-1}) = -1 + g_1z^{-1} + \ldots + g_{k-1}z^{-(k-1)}$$

$$F(z^{-1}) = f_o + f_1z^{-1} + \ldots + f_{s-1}z^{-(s-1)}$$

$$s = \max(n_a + n_{P_d},\ n_c + n_{P_n} - k + 1)$$

Combining Equation 3.18 and 3.19 then substituting for $\xi(t)$ from Equation 3.16 results in the following equation for the predicted weighted control error:

$$P\epsilon(t+k) = \frac{F}{P_dC}\epsilon(t) + \frac{BE}{C}\eta(t) + E\xi(t+k) \qquad (3.20)$$

The optimal prediction of $P\epsilon(t+k)$, if $\xi(t)$ is assumed to be zero mean, is given by:

$$P\epsilon^*(t+k|t) = \frac{F}{P_dC}\epsilon(t) + \frac{BE}{C}\eta(t) \qquad (3.21)$$

Substituting the optimal prediction into the performance index and completing the minimization results in:

$$P\epsilon^*(t+k|t) + Q[\eta(t) + P\epsilon(t)] = 0 \qquad (3.22)$$

where

$$Q = -(q'_{n_o}/q'_{d_o})(Q'/b_o)$$

The general controller can then be obtained by substituting Equations 3.21 and 3.22 into Equation 3.13. For minimum variance type control, i.e. $P(z^{-1}) = 1$ and $Q(z^{-1}) = 0$, the auxiliary signal $\eta(t)$ is given by:

$$\eta(t) = \frac{-F(z^{-1})}{B(z^{-1})E(z^{-1})}\epsilon(t) \qquad (3.23)$$

The robust minimum variance control law then becomes:

$$u(t) = \frac{1}{D}\left[1 - \frac{F}{BE}\right]\epsilon(t) \qquad (3.24)$$

3.2.3 Simplification to a Self-Tuning PID Form

A self-tuning PID controller can be derived from the general SRC if the following conditions are imposed.

1. The process can be modelled by the following second order ARIMA model

$$A_m(z^{-1})y(t) = z^{-k}B_m(z^{-1})u(t) + H_m(z^{-1})w(t) + C_m(z^{-1})\xi(t)/\Delta \qquad (3.25)$$

where

$$\begin{aligned} A_m(z^{-1}) &= 1 + a_1z^{-1} + a_2z^{-2} \\ B_m(z^{-1}) &= b_o \\ C_m(z^{-1}) &= 1 \\ k &= 1 \quad \text{(sampling delay)} \end{aligned} \qquad (3.26)$$

2. The external signals can be characterized by step like changes, i.e.:

$$D(z^{-1}) = 1 - z^{-1} \qquad (3.27)$$

3. The performance index weighting transfer functions are chosen to be $P(z^{-1}) = 1$ and $Q(z^{-1}) = 0$

Remark: Strictly speaking this self-tuning PID control law is an approximation of the minimum variance based controller since $Q(z^{-1})=0$. However, in later chapters the GMV qualifier will be used as a label when referring to this PID control law since it designates the general design principle from which it is derived.

Based on the above conditions the auxiliary signal, $\eta(t)$, becomes:

$$\eta(t) = \frac{f_o + f_1 z^{-1} + f_2 z^{-2}}{b_o} \epsilon(t) \tag{3.28}$$

The resulting control law then follows as:

$$u(t) = \frac{(b_o+f_o) + f_1 z^{-1} + f_2 z^{-2}}{b_o(1 - z^{-1})} \epsilon(t) \tag{3.29}$$

This 3 term control law can be cast into a conventional discrete control algorithm where the output error drives the proportional, integral and derivative terms. Equation 3.29 can be represented as the fixed discrete PID algorithm of Equation 2.8 with:

$$\begin{aligned} K_c &= \frac{-1}{b_o}(f_1 + 2f_2) \\ T_i &= \frac{-(f_1 + 2f_2)T_s}{b_o + f_o + f_1 + f_2} \\ T_d &= \frac{-f_2 T_s}{f_1 + 2f_2} \end{aligned} \tag{3.30}$$

The regression model is implicit with the structure:

$$\epsilon(t) = F\epsilon(t-1) + B\eta(t-1) + \xi(t) \tag{3.31}$$

Both the data vector and the measurement are zero mean and the noise is uncorrelated with the data.

This controller can be reformulated so that explicit, rather than implicit, estimation of the plant parameters is carried out. The Diophantine identity specific to the above conditions is of the form:

$$1 = (A_m D + B_m z^{-1}) + Fz^{-1} \tag{3.32}$$

This equation results from substituting Equations 3.15 and 3.26 into Equation 3.19 with $P(z^{-1})=1$. Solving for the f_i parameters in terms of a_i and b_o, Equations 3.28 becomes:

$$u(t) = \frac{(1-a_1) + (a_1-a_2)z^{-1} + a_2 z^{-2}}{b_o(1 - z^{-1})} \epsilon(t) \tag{3.33}$$

Reworking into the form of Equation 2.8, the controller parameters are then:

$$\begin{aligned} K_c &= \frac{-1}{b_o}(a_1 + a_2) \\ T_i &= -(a_1 + a_2)T_s \\ T_d &= \frac{-a_2 T_s}{a_1 + a_2} \end{aligned} \tag{3.34}$$

and the explicit regression model becomes:

$$\Delta y(t) = bo\Delta u(t-1) - a1\Delta y(t-1) - a2\Delta y(t-2) + \xi(t) \tag{3.35}$$

Again the data vector and measurement have a zero mean and the noise is uncorrelated with the data vector.

3.3 Integrating Self-Tuning Controller

3.3.1 Introduction

Interest in self-tuning control began with the groundbreaking work of Åström and Wittenmark (1973). In it the self-tuning regulator (STR) was developed which in the limiting case was a minimum variance controller. In 1975, and later in 1979, Clarke and Gawthrop introduced their self-tuning controller (STC). This took the form of a generalized minimum variance controller which allowed for setpoint, control and output weighting. This STC has been modified in various ways since 1979. Clarke et al. (1983) proposed the k-incremental STC to handle the offset problem. Tuffs and Clarke (1985) developed the integrating form of the Clarke-Gawthrop self-tuner as a unified approach to handle offsets in STC's.

This integrating STC is based on a ARIMA process model representation. It is a development of this controller which will be used to derive a self-tuning regulator in PID form. For more details of the original STC derivation the reader is referred to the earlier works of Tuffs (1985) and Tuffs and Clarke (1985).

3.3.2 Derivation of the Integrating Form of the Self-Tuning Controller

The plant model is chosen as the ARIMA model as set out in Equation 3.36 with $C(z^{-1}) = 1$. This is not required but simplifies the algebra that follows.

$$A(z^{-1})y(t) = z^{-k}B(z^{-1})u(t) + \xi(t)/\Delta \qquad (3.36)$$

The control law is defined as:

$$G(z^{-1})\Delta u(t) + F'(z^{-1})y(t) - H(z^{-1})y_r(t) = 0 \qquad (3.37)$$

where $G(z^{-1})$, $F'(z^{-1})$ and $H(z^{-1})$ are polynomials in z^{-1}. The above control law is derived by minimizing the following cost function with respect to u(t):

$$J = E\{[Py(t+k) - Hy_r(t)]^2 + [Q'u(t)]^2\} \qquad (3.38)$$

where $P(z^{-1}) = 1$ and $Q(z^{-1}) = 0$ and the ARIMA system model is assumed. The detailed derivation of the control law is very similiar to Section 3.2 and is omitted here.

Combining Equations 3.36 and 3.37 results in the following closed-loop equation:

$$(AG\Delta + z^{-k}BF')y(t) = z^{-k}BHy_r(t) + G\xi(t) \qquad (3.39)$$

A Diophantine equation which includes the weighting polynomial $P(z^{-1})$ can be written as:

$$PB = AG\Delta + z^{-k}BF' \qquad (3.40)$$

By defining G = EB, Equation 3.40 becomes:

$$P = AE\Delta + z^{-k}F' \qquad (3.41)$$

Substitution of Equation 3.41 into 3.39 gives the equivalent closed-loop expression:

$$Py(t) = Hy_r t-k) + E\xi(t) \qquad (3.42)$$

Implementation of the controller defined in Equation 3.37 requires the polynomials $G(z^{-1})$, $F'(z^{-1})$ and $H(z^{-1})$ be specified. A regression model can be formulated to estimate $G(z^{-1})$ and $F'(z^{-1})$ by multipling Equation 3.36 by $E(z^{-1})$ and substituting Equation 3.41:

$$Py(t) = G\Delta u(t-k) + F'y(t-k) + E\xi(t) \qquad (3.43)$$

Equation 3.43 can be further modified by defining:

$$F'(z^{-1}) = 1 + \Delta F(z^{-1}) \qquad (3.44)$$

The resulting regression model is now:

$$Py(t) - y(t-k) = G\Delta u(t-k) + F\Delta y(t-k) + E\xi(t) \qquad (3.45)$$

This regression model has zero-mean data and measurement and the noise is uncorrelated with the data vector (i.e. $DegE=k-1$).

3.3.3 Simplification to a Self-Tuning PID Form

A self-tuning PID controller can be obtained from the integrating STC if the following conditions are imposed:

1. The process can be modelled by a second order ARIMA system model with

$$A(z^{-1}) = 1 + a_1 z^{-1} + a_2 z^{-2}$$
$$B(z^{-1}) = b_o \qquad (3.46)$$

$k = 1$ (sampling delay)

2. The cost function weighting polynomials are chosen to be $P(z^{-1}) = 1$, $Q(z^{-1}) = 0$ and $H(z^{-1}) = 1$

Remark: Strictly speaking this self-tuning PID control law is an approximation of the minimum variance based controller since $Q(z^{-1})=0$. However, in later chapters the GMV qualifier will be used as a label when referring to this PID control law since it designates the general design principle from which it is derived.

Based upon these conditions the degree of polynomial $F(z^{-1})$ is DegF=1 and the degree of polynomial $E(z^{-1})$ is DegE=0. The implicit 3-term control law then becomes:

$$u(t) = \frac{y_r(t) - y(t) - \Delta(f_o+f_1z^{-1})y(t)}{g_o(1 - z^{-1})} \tag{3.47}$$

This control law can be cast into a velocity "setpoint-on-I-only" PID control algorithm form as seen in Equation 2.10. The conventional controller parameters are calculated as:

$$\begin{aligned} K_c &= \frac{1}{g_o}(f_o + f_1) \\ T_i &= (f_o + f_1)T_s \\ T_d &= \frac{-f_1T_s}{f_o + f_1} \end{aligned} \tag{3.48}$$

The regression model for the implicit formulation is:

$$\Delta y(t) = G\Delta u(t-1) + F\Delta y(t-1) + E\xi(t) \tag{3.49}$$

This controller can be reformulated in explicit form by solving the Diophantine identity below:

$$1 = AE\Delta + z^{-1}(1 + \Delta F) \tag{3.50}$$

Solving for the f_i and g_o parameters in terms of a_i and b_o, the following control law can be derived:

$$u(t) = \frac{y_r(t) - y(t) + \Delta(a_1 + a_2 z^{-1})y(t)}{b_o(1 - z^{-1})} \tag{3.51}$$

Casting Equation 3.51 into the form of Equation 2.10, the controller constants take the form:

$$\begin{aligned} K_c &= \frac{-1}{b_o}(a_1 + a_2) \\ T_i &= -(a_1 + a_2)T_s \\ T_d &= \frac{-a_2 T_s}{a_1 + a_2} \end{aligned} \tag{3.52}$$

and the explicit regresion model is:

$$\Delta y(t) = b_o\Delta u(t-1) - a_1\Delta y(t-1) - a_2\Delta y(t-2) + \xi(t) \tag{3.53}$$

3.4 Self-Tuning Pole Placement Controller

3.4.1 Introduction

The design of self-tuning controllers based on pole-zero placement methods has been considered by many authors in the literature. The general idea has been popularized by Wellstead and co-workers (1979a, 1979b, 1981) and Åström and Wittenmark (1980).

Wittenmark and Åström (1980) and Isermann (1981) have used the pole placement method to design "simple" self-tuning controllers. Some of these controllers can lead to three-term controllers.

The particular pole placement controller to be summarized here is the one proposed by Tjokro (1984) and Tjokro and Shah (1985) based on the ARIMA model. The same format of development has been used and the reader is referred to the original references for a more detailed derivation.

3.4.2 Derivation of the Self-Tuning Pole Placement Controller

Consider a single-input, single-output process which can be characterized by a linear ARIMA model:

$$A(z^{-1})y(t) = z^{-k}B(z^{-1})u(t) + C(z^{-1})\xi(t)/\Delta \qquad (3.54)$$

where $A(z^{-1})$, $B(z^{-1})$ and $C(z^{-1})$ are polynomials in the backward shift operator z^{-1}, $A(z^{-1})$ is monic and k is the system time delay ($k \geq 1$). The process input and output are u(t) and y(t) respectively. $N(t)=C(z^{-1})\xi(t)/\Delta$ is the residual which accounts for the effects of unmeasured disturbances, noise, modelling error and non-linearities. $C(z^{-1})$ is assumed to be unity.

Consider a general linear feedback controller of the form:

$$G(z^{-1})\Delta u(t) = H(z^{-1})y_r(t) - F(z^{-1})y(t) \qquad (3.55)$$

where $F(z^{-1})$, $G(z^{-1})$ and $H(z^{-1})$ are polynomials in z^{-1}. $G(z^{-1})$ is monic and $y_r(t)$ is the process reference signal.

Combining Equations 3.54 and 3.55 results in the closed-loop transfer function:

$$y(t) = \frac{z^{-k}BHy_r(t) + G\xi(t)}{(AG\Delta + z^{-k}BF)} \qquad (3.56)$$

According to pole placement criterion, the coefficients of the polynomials F(z-1), $G(z^{-1})$ and $H(z^{-1})$ are determined in such a way that the closed-loop system has the following desired closed-loop transfer function:

$$y(t) = \frac{X(z^{-1})}{W(z^{-1})}y_r(t) + \frac{S(z^{-1})}{W(z^{-1})}\xi(t)/\Delta \qquad (3.57)$$

where $S(z^{-1})$, $W(z^{-1})$ and $X(z^{-1})$ are polynomials in z^{-1} with $W(z^{-1})$ being monic. These are pre-specified by the designer. $W(z^{-1})$ is the desired closed-loop characteristic polynomial.

Equating the coefficients of $y_r(t)$ and $\xi(t)$ in Equations 3.56 and 3.57 gives:

$$AG\Delta + z^{-k}BF = \frac{GW\Delta}{S} \qquad (3.58)$$

$$H = \frac{GX\Delta}{z^{-k}BS} \qquad (3.59)$$

The controller design problem has now been reduced to the solution of Equations 3.58 and 3.59 to find the controller polynomials $F(z^{-1})$, $G(z^{-1})$ and $H(z^{-1})$, provided that polynomials $A(z^{-1})$ and $B(z^{-1})$ are known and polynomials $S(z^{-1})$, $W(z^{-1})$ and $X(z^{-1})$ are pre-specified. The polynomials $A(z^{-1})$ and $B(z^{-1})$ can be determined by recursive parameter estimation based on the following explicit incremental regression model:

$$A(z^{-1})\Delta y(t) = B(z^{-1})\Delta u(t-k) + \xi(t) \qquad (3.60)$$

This regression model has zero-mean data and measurement in the steady state and the noise is uncorrelated with the data vector.

3.4.3 Simplification to a Self-Tuning PID Form

A self-tuning PID algorithm can be derived from this self-tuning pole placement controller if the following conditions are imposed:

1. The process can be represented by at most a second order plus time delay ARIMA model with:

$$A(z^{-1}) = 1 + a_1z^{-1} + a_2z^{-2}$$
$$B(z^{-1}) = b_o + b_1z^{-1} \qquad (3.61)$$
$$C(z^{-1}) = 1$$

2. A steady-state approximation for the numerator term can be used such that:

$$z^{-k}B(z^{-1}) = \Sigma b_i \qquad (3.62)$$

3. The desired closed-loop characteristic polynomial $W(z^{-1})$ is chosen as a first order polynomial, i.e.

$$W(z^{-1}) = 1 + w_1z^{-1} \qquad (3.63)$$

4. The design polynomials $S(z^{-1})$ and $X(z^{-1})$ are specified to be:

$$S(z^{-1}) = \Delta G(z^{-1})/(K_1A(z^{-1})) \qquad (3.64)$$
$$X(z^{-1}) = K_2z^{-k}B(z^{-1})/K_1 \qquad (3.65)$$

where

$K_1 = z^{-k}\Sigma b_i$

$K_2 = 1 + w_1$

5. The controller polynomial $F(z^{-1})$ is fixed as:

$$F(z^{-1}) = K_2A(z^{-1}) \qquad (3.66)$$

Based on the above conditions, the control law can be expressed as (Tjokro, 1984):

$$u(t) = u(t-1) + \lambda[\epsilon(t) + a_1\epsilon(t-1) + a_2\epsilon(t-2)] \qquad (3.67)$$

where

$$\lambda = -(1+w_1)/(w_1\Sigma b_i) \qquad (3.68)$$

$$\epsilon(t) = y_r(t) - y(t) \qquad (3.69)$$

This controller can be shown to be equivalent to a discrete velocity PID algorithm with "setpoint-on-P&I&D" form as shown in Equation 2.8 with:

$$K_c = \frac{-(1+w_1)(a_1 + 2a_2)}{w_1\Sigma b_i}$$

$$T_i = \frac{-(a_1 + 2a_2)T_s}{1 + a_1 + a_2} \qquad (3.70)$$

$$T_d = \frac{-a_2T_s}{a_1 + 2a_2}$$

If the additional steady-state approximation of

$$A(z^{-1})y_r(t) = (1+\Sigma a_i)y_r(t) \qquad (3.71)$$

is made, a velocity PID control law with "setpoint-on-I-only" form similiar to Equation 2.10 can be derived.

The incremental regression model for the self-tuning PID controller is then:

$$\Delta y(t) = B(z^{-1})\Delta u(t-k) - a_1\Delta y(t-1) - a_2\Delta y(t-2) + \xi(t) \qquad (3.72)$$

3.5 Comparison of Self-Tuning Control PID Algorithms

In the previous sections three self-tuning PID algorithms are developed by reduction from their more complex "complete" self-tuning counterparts which were derived using generalized minimum variance (GMV) and pole placement (PP) criteria. These algorithms are summarized and compared in Table 3.1.

The regression model for each controller is based on incremental variables, a result of the ARIMA system representation. Both the SRC and integrating STC derived self-tuning PI(D) controllers can be formulated with implicit and explicit parameter regression model. The implicit formulation is how the SRC and integrating STC are presented in the literature. The explicit forms for both controllers is derived in Sections 3.2 and 3.3. It is interesting to note that the explicit SRC and STC PID controllers calculate identical K_c, T_i and T_d. However, the SRC PID controller has a "setpoint-on-P&I&D" structure while the integrating STC has a "setpoint-on-I-only" structure.

The controller gain, K_c, is a function of all the estimated parameters and for the pole placement PID algorithm w_1. The desired closed-loop pole w_1 actually becomes a scaling factor for K_c. The integral and derivative times are a function of the parameters corresponding to the process output y(t) and the sampling time, T_s. For the explicit form this corresponds to the a_i parameters.

Table 3.1 Comparison of self-tuning PID algorithms

Control Law	Regression Model	K_c	T_i	T_d
SRC	Implicit Incremental	$\frac{-(f_1+2f_2)}{b_0}$	$\frac{-(f_1+2f_2)T_s}{b_0+f_0+f_1+f_2}$	$\frac{-f_2T_s}{f_1+2f_2}$
SRC	Explicit Incremental	$\frac{-(a_1+a_2)}{b_0}$	$-(a_1+a_2)T_s$	$\frac{-a_2T_s}{a_1+a_2}$
STC	Implicit Incremental	$\frac{(f_0+f_1)}{g_0}$	$(f_0+f_1)T_s$	$\frac{-f_1T_s}{f_0+f_1}$
STC	Explicit Incremental	$\frac{-(a_1+a_2)}{b_0}$	$-(a_1+a_2)T_s$	$\frac{-a_2T_s}{a_1+a_2}$
PP	Explicit Incremental	$\frac{-(1+w_1)(a_1+2a_2)}{w_1\Sigma b_i}$	$\frac{-(a_1+2a_2)T_s}{1+a_1+a_2}$	$\frac{-a_2T_s}{a_1+2a_2}$

3.6 Alternate Method for Controller Gain Calculation

The three self-tuning PID control algorithms calculate K_c as summarized in Table 3.1. The SRC and integrating STC PID controller gain is totally determined by the estimated parameters. No additional tuning of K_c is allowed by the equations. The pole placement PID controller K_c calculation includes a factor w_1 which allows the gain calculation to be scaled if necessary. In essence w_1 is a tuning knob. Due to the simplifying assumptions to obtain a PID controller structure, this tuning knob has only an indirect relationship to any characteristic of the closed-loop system. An alternate method for K_c calculation is now proposed which is directly related to the transient behavior of the closed-loop. In the ideal case the alternate method exactly specifies the maximum overshoot (M_p) of the closed-loop system when subjected to a step change in the reference signal. The method determines K_c by locating the closed-loop poles in locations to achieve the desired transient reponse. The proposed method is described in the following section.

Consider a continuous second order process with no numerator dynamics:

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad (3.73)$$

where s is the complex Laplace operator, ζ is the damping ratio and ω_n is the natural frequency of the process. If

$\zeta>1$ the process is overdamped with real and distinct poles. If $\zeta=1$ the process is critically damped with real and equal poles. If $\zeta<1$ the process is underdamped with complex conjugate poles. The maximum overshoot (M_p) is directly related to the damping ratio (Coughanowr and Koppel, 1965):

$$\begin{aligned}&\text{If } \zeta\geq 1, \text{ then } M_p = 0\\&\text{If } \zeta<1, \text{ then } M_p = \mathrm{EXP}(-\pi\zeta/(1-\zeta^2)^{0.5})\end{aligned} \tag{3.74}$$

Consider now a discrete deterministic second order process which can be viewed as the discrete equivalent of Equation 3.73:

$$A(z^{-1})y(t) = z^{-1}B(z^{-1})u(t) \tag{3.75}$$

where

$$A(z^{-1}) = \cancel{1 + a_1z}^{-1} + a_2z^{-2}$$
$$B(z^{-1}) = b_0 + b_1z^{-1}$$

The self-tuning PID controller designed for this process has the form:

$$\Delta u(t) = K_cN(z^{-1})\epsilon(t) \tag{3.76}$$

where

$$N(z^{-1}) = (1 + \frac{T_s}{T_i} + \frac{T_d}{T_s}) - (1 + 2\frac{T_d}{T_s})z^{-1} + \frac{T_d}{T_s}z^{-2}$$

From the specified process model and PID controller the closed-loop characteristic polynomial is derived and has the form:

$$A(z^{-1})\Delta + z^{-1}K_cB(z^{-1})N(z^{-1}) = 0 \qquad (3.77)$$

The degree of this polynomial is fourth order (i.e. four closed-loop poles). If the self-tuning PID controller is derived from the pole placement method, Equation 3.77 is effectively second order since T_i and T_d take on values so that the controller cancels the estimated process poles (i.e. $N(z^{-1})=\alpha A(z^{-1})$). Therefore, there are two closed-loop poles which determine the dominant closed-loop system response characteristics. The effective characteristic polynomial is now:

$$\Delta + z^{-1}\alpha K_cB(z^{-1}) \qquad (3.78)$$

where

$$\alpha = -1 / (a_1 + 2a_2)$$

An analysis of Equation 3.78 reveals that the closed-loop poles originate at the open-loop poles of Δ and terminate at the open-loop poles of $B(z^{-1})$. The exact location of the closed-loop poles is determined by K_c and can be related to maximum overshoot.

Consider a particular choice of K_c. The closed-loop poles are determined by solving the quadratic Equation 3.78. The damping ratio is calculated from the closed-loop pole locations in the complex z-plane according to the following formulae (Phillips and Nagle, 1984):

Stable real positive poles

$$\zeta = 1.0 \qquad (3.79)$$

Stable complex conjugate poles
Stable real negative poles

$$\zeta = -\ln r \,/\, (\ln^2 r + \theta^2)^{0.5} \qquad (3.80)$$

where r and θ are the polar coordinates of the closed-loop poles. Franklin and Powell (1980) show Equation 3.80 graphically as concentric logarithmic spirals. This damping ratio can be related to M_p via Equations 3.74 and K_c incremented accordingly to reach the M_p setpoint.

The qualitative algorithm for the alternate K_c calculation is summarized below:

1. Specify desired maximum overshoot (M_p) setpoint (i.e. $M_p^{sp}=15$)
2. Setup the current estimated characteristic polynomial and solve for dominant closed-loop pole locations.
3. Calculate the damping ratio coresponding to the dominant closed-loop pole locations.
4. Calculate equivalent M_p
5. Make appropriate adjustment in K_c so that M_p equals M_p^{sp}.

```
IF ($M_p > M_p^{sp}$) THEN
    $K_c = K_c - K_c$Increment
ELSEIF ($M_p < M_p^{sp}$) THEN
    $K_c = K_c + K_c$Increment
ENDIF
```

4. Additional Techniques for Enhanced Self-Tuning Control

4.1 Introduction

The effectiveness of an adaptive or self-tuning controller is not only contingent upon the strucure and parameters of the control algorithm itself but also on the steps that proceed the actual calculation of the control law output. A self-tuning controller is a complex, nonlinear, time-varying controller. It is necessary to determine the process structure, estimate the process model parameters and carry out controller design calculations automatically. The practical application of these controllers requires the engineer to make certain *a priori* decisions such as choice of sampling time, recursive parameter estimation techniques, model reduction and filtering techniques, inclusion of feedforward and time delay compensation. These are the important issues that will be considered in this chapter.

4.2 Choice of Sampling Period

One of the first decisions to be made in digital control is the rate at which the process is to be sampled and control implemented. The sampling time, T_s, greatly influences the behavior of the controller and therefore the closed-loop response. The importance of the choice of sampling period is not specific only to adaptive control but is an important design parameter for all sampled-data design methods (Wittenmark and Åström, 1984). The particular choice of T_s varies greatly depending on the application and the specified performance criteria. Some of the basic considerations in choosing an appropriate sampling period are summarized by the following points:

1. The sampling rate must be fast enough so that significant process information is not lost and the reconstruction of the continuous-time signal is possible. Shannon's sampling theorem (Shannon, 1949) states that a sinusoidal signal must be sampled at least twice each period if the original signal is to be recovered. Process signals of interest in the process industries are typically not sinusoidal. Shannon's theorem translates into identifying the highest frequency of importance and sampling the signal at twice that frequency (Seborg, Edgar and Mellichamp, 1983).
2. A fast sample rate allows one to use the parameters of the continuous-time controller directly in a discrete conventional PID controller. It must be fast in order to obtain a good approximation of the derivative of the output (Wittenmark, 1979). Sampling too slowly can reduce the effectiveness of the feedback control system to load disturbances and setpoint changes since a sampled-data system runs effectively open-loop between sample periods (Åström, 1985). The above requirements must be compromised with the increased load imposed on the central computer by fast sampling conditions.

3. Sampling at a fast rate can cause numerical problems for the parameter estimator. With a small T_s the b_i parameters tend to become very small. If the control law uses these parameters in the denominator of the control law calculation, the amplitude of the manipulated variable can become very large (Isermann, 1986).
4. For processes with time delays, it is desireable that the sample period be chosen so that the time delay be expressed as an integer multiple of the sample period. If this is not the case, a fractional delay may give rise to a non-minimum phase zero in the discrete-time domain (Wellstead et al., 1979; Goodwin and Sin, 1984).
5. The sampling rate at which a continuous plant is sampled largely determines the location of the discrete system poles and zeroes. The continuous pole, p_s, is mapped into the z-plane by $e^{p_s T_s}$. A stable, real, continuous pole is translated into the real axis of the z-plane between 0 and 1. At high sample rates the poles tend toward z=1 and at low rates towards z=0. Phillips and Nagle (1984) summarize this mapping of poles from the s-plane to the z-plane.

There is no simple transformation for zeroes. It is possible to have a nonstably invertible discrete-time model from a stably invertible continuous-time model. The converse is also true - to have a stably invertible discrete-time model from a nonstably invertible continuous-time model (Åström, Hagander, and Sternby, 1984). It can be shown that all continuous-time systems with pole excess greater than 2 will give sampled-data systems with non-minimum phase zeroes if the sampling period is sufficiently small. It is therefore very comnmon to have unstable zeroes in discrete-time systems(Åström et al., 1984). As well a non-minimum phase continuous-time model can yield a minimum phase discrete-time plant if the sample period is sufficiently long. Therefore, caution must be shown in the choice of sample rate when designing controllers which rely on inverse stability of the system. Caution must also be shown in cancelling the ringing zeroes of a process with the control law as this will result in ringing of the control signal (Wittenmark and Åström, 1984).

6. The effects of process/model mismatch can be reduced by a proper choice of sampling rate. Longer sampling periods are good for limiting the bandwidth of the controller and therefore minimizing the effects of unmodelled higher-frequency dynamics (Cluett, Shah and Fisher, 1986; Rohrs et al.,1984). Åström (1985) states that the choice of sample period is critical when a low order model is fitted to a higher order process. The quality of the approximation is increased if the sample period is increased.

7. From a regulation point of view it is desireable to have a fast sample rate. However, from a discrete-time model estimation view it is desireable to have a slower sample rate. These citeria may be satisfied if an asynchronous combination is adopted (Isermann, 1986). This strategy would update the parameter estimates over several sample periods while computing new control action every sampling instant based on the most current parameter estimates (Goodwin and Sin, 1984).
8. Process and measurement noise can be attenuated by proper selection of the sampling rate. At low signal-to-noise ratios rapid sampling should be avoided since the measurement will primarily show the effects of the noise. A high frequency roll-off can be obtained with lower sample rates. If a high sample rate is required for control performance, the process signal should be filtered by an effective anti-aliasing filter such as an analog exponential filter (Wittenmark and Åström, 1984; Seborg et al., 1987).

Many of these design considerations have been applied in explicit criteria for determining an appropriate sample rate for a process. Seborg et al. (1987) has summarized in tabular form some general guidelines for selecting the sampling period for discrete PID controllers based on methods published in the literature. The methods vary as to their basis and criteria. For an open-loop system the sample period can be determined from the dominant time constant, response time and the critical frequency of the continuous system. The sampling time (T_s) can also be based on the bandwidth or rise time of the desired closed-loop response.

For the purposes of this study the open-loop response time method proposed by Isermann (1982) has been used. It is of the form

$$\frac{t_r}{15} < T_s < \frac{t_r}{6} \tag{4.1}$$

where T_s is the sampling period and t_r is the 95% complete response time of the open-loop system. This was found to locate the discrete-time system poles in an acceptable location in the unit circle (i.e. $0.3 < \text{pole} < 0.9$) and to allow the controller to give acceptable performance.

4.3 Recursive Parameter Estimation

The on-line estimation of unknown process and controller parameters is at the core of adaptive and self-tuning controllers. The body of literature devoted to this topic is extensive and has grown considerably over the last 15 years (for example, Åström and Eykhoff, 1971; Strejc, 1980; Isermann, 1980). The aim of this section is to document the recursive algorithm and implementation used in this work and discuss other related issues.

4.3.1 Parameter Estimation Methods

Some possible choices of recursive algorithms for parameter estimation of linear systems are : recursive projection algorithm (RPA), recursive least squares (RLS), recursive extended least squares (RELS), recursive maximum likelihood (RML) and recursive instrumental variables (RIV). The actual choice is governed by considerations of convergence, computation effort, sensitivity to nonzero process bias and performance in the presence of process and measurement noise.

Equation 3.6 describes the ARIMA system representation adopted in this thesis. The regression model corresponding to representation is:

$$\Delta y(t) = (1-A(z^{-1}))\Delta y(t) + B(z^{-1})\Delta u(t-k) + C(z^{-1})\xi(t) \qquad (4.2)$$

The regression model differences the input and output data as a result of the assumed integrating noise model. If a special noise model of $N(t)=\xi(t)/(A(z^{-1})\Delta)$ (i.e. $C(z^{-1})=1$) is presumed, recursive least squares can be used to identified the process parameters (Isermann, 1980). The algorithm is then of the form:

$$\theta(t) = \theta(t-1) + K(t)\hat{e}(t|t-1) \qquad (4.3)$$

$$K(t) = P(t-1)\phi(t)[\lambda(t) + \phi^T(t)P(t-1)\phi(t)]^{-1} \qquad (4.4)$$

$$P(t) = [I - K(t)\phi(t)]\, P(t-1)/\lambda(t) \qquad (4.5)$$

$$\hat{e}(t|t-1) = \Delta y(t) - \phi^T(t)\theta(t-1) \tag{4.6}$$

where K(t) is the Kalman gain, $\hat{e}(t|t-1)$ is the *a priori* incremental prediction error, P(t) is the covariance matrix and $\lambda(t)$ the forgetting factor (to be defined later). The parameter vector and input-output regressor vector are $\theta(t)$ and $\phi(t)$ respectively and are defined as:

$$\theta(t) = [\ \hat{a}_1 \ldots \hat{a}_{n_a} ; \hat{b}_0 \ldots \hat{b}_{n_b}]^T \tag{4.7}$$

$$\begin{aligned}\phi(t) = [&-\Delta y(t-1) \ldots -\Delta y(t-n_a); \\ &\Delta u(t-k) \ldots \Delta u(t-k-n_b)]^T\end{aligned} \tag{4.8}$$

where n_a and n_b is the order of the polynomials $A(z^{-1})$ and $B(z^{-1})$respectively. The RLS algorithm has the advantage of a relatively small computation expense, reliable convergence. It has the disadvantages of giving biased parameter estimates if the assumed noise model differs significantly fron the true noise model (Isermann, 1982).

If the process noise can be modelled realistically by $N(t)=C(z^{-1})\xi(t)/(A(z^{-1})\Delta)$, the recursive extended least squares method or maximum likelihood method can be used to estimate the process and noise parameters. If RELS is selected, the basic algorithm is similiar to Equations 4.3-4.6. The main difference being that the parameter vector and the regressor vector are augmented to estimate the $C(z^{-1})$ polynomial parameters as shown below:

$$\theta(t) = [\hat{a}_1 \ldots \hat{a}_{n_a} ; \hat{b}_0 \ldots \hat{b}_{n_b} ; \hat{c}_1 \ldots \hat{c}_{n_c}]^T \tag{4.9}$$

$$\phi_b(t) = [-\Delta y(t-1) \ldots -\Delta y(t-n_a); \Delta u(t-k) \ldots \Delta u(t-k-n_b); \hat{e}_b(t-1|t-1) \ldots \hat{e}_b(t-n_c|t-n_c)]^T \quad (4.10)$$

The particular RELS method used in this work is the one proposed by Young (1972) and summarized in Strejc (1980). This modified version uses the most recent estimate of the parameters in calculating the prediction error for insertion into the input-output (I/O) regressor shown in Equation 4.10. The two modified prediction error equations tnen become:

$$\hat{e}_b(t|t-1) = \Delta y(t) - \phi_b^T(t)\theta(t-1) \quad (4.11)$$

$$\hat{e}_b(t|t) = \Delta y(t) - \phi_b^T(t)\theta(t) \quad (4.12)$$

The *a priori* prediction error of Equation 4.11 is used for the parameter update (Equation 4.3) and the *a posteriori* prediction error of Equation 4.12 is used in the I/O regressor. Performance of RELS is good for this special noise model. Computational expense is marginally greater than that for the RLS and convergence of the $\hat{c}_i$ parameters is slower than the process parameters. It is recommended that RELS only be used for large amplitude stationary noise (Isermann, 1986).

If a noise model $N(t)=C(z^{-1})\xi(t)/(D(z^{-1})\Delta)$ is encountered where $D(z^{-1})$ is a general monic polynomial, parameter estimation algorithms like the recursive instrumental variables method (RIV) is recommended to obtain unbiased parameter estimates of the system polynomials. In this work the noise model will be assumed to be of the form $N(t)=C(z^{-1})\xi(t)/(A(z^{-1})\Delta)$, such that an RELS estimator will be adequate for parameter estimation.

4.3.2 Discounting Old Data

One of the key properties of an adaptive controller is the ability to track changes in the process dynamics. To do this effectively it is necessary to discount old data. The rate at which the data is discounted is of significant importance. Discounting too rapidly causes covariance matrix windup if the process is at steady state whilst too slow discounting renders the adaptive controller unable to follow rapid parameter variations (Wittenmark and Åström, 1984).

One way to discount old data is to use exponential forgetting. This is typically done by the use of a forgetting factor, λ. If $\lambda=1$ all data is weighted equally. If $\lambda<1$ more recent data is weighted more than old data. The smaller λ is, the greater the discounting of old data. The asymtotic sample length (ASL) of the parameter estimation algorithm is calculated as:

$$ASL = 1 / (1 - \lambda) \tag{4.13}$$

The use of exponential data forgetting allows rapidly changing parameters to be tracked, but it can also cause problems. If there are long periods of almost quiescent operation, i.e. the process is insufficiently excited, problems of covariance windup and parameter bursting are encountered. If $\lambda<1$ in the absence of process excitation, parameter uncertainty increases and the elements of the covariance matrix grow exponentially. This is known as estimator windup. Under such conditions when new information suddenly enters the process (for example, a process disturbance), a numerically large covariance matrix can result in wildly varying parameter variations - parameter bursting.

These problems can be handled by properly exciting the process or eliminating the discounting during periods of low excitation. This second method has give rise to variable forgetting factor schemes (Fortescue et al., 1981; Morris et al., 1977; Hägglund, 1983). The particular variable forgetting factor scheme used in this work is one presented by Montague et al. (1986):

$$\lambda_1(t) = a\lambda_1(t-1) + (1-a)\lambda_f \tag{4.14}$$

$$\lambda_2(t) = 1.0 - \frac{K_1 * \hat{e}(t|t-1)^2}{1.0 - K_2\hat{e}(t|t-1)^2} \tag{4.15}$$

$$\lambda(t) = \lambda_1(t) * \lambda_2(t) \tag{4.16}$$

where $\lambda(t)$ is the variable forgetting factor, a is a filter constant, λ_f is the forgetting factor value at steady state, K_1 and K_2 are constants and $\hat{e}(t|t-1)$ is the *a priori* incremental prediction error. Furthermore, $\lambda(t)$ is set to λ_{min} if $\lambda(t)<\lambda_{min}$ and $\lambda(t)=\lambda_f$ if $\lambda(t)>\lambda_f$. This variable forgetting factor method is a simple approach but it does require the selection of four parameters. For future work the more rigorous variable forgetting factor method of Ydstie et al. (1985) is recommended.

4.3.3 Estimation Algorithm Conditioning

Potential numerical conditioning problems of the recursive algorithm may be encountered during the parameter estimation step of self-tuning control. Åström (1983) states that the parameter estimation algorithm can be poorly conditioned (Hanson and Lawson, 1969) for any of the following reasons:

1. Overparameterization of the assumed process models.
2. Lack of process excitation. The least squares algorithm is poorly conditioned to high signal-to-noise ratios.
3. Signals with high superimposed nonzero mean data levels.
4. Finite word length effects in microcomputers.

These conditioning and the consequential numerical problems can be reduced by minimizing the above conditions. Of particular importance is removal of the nonzero mean data level and use of factorization algorithms to implement the recursive estimation method. Factorization algorithms (for example, Bierman and Thorton, 1976; Strejc, 1980; Radke, 1984) update a factored covariance matrix instead of the full covariance matrix so as to maintain the positive definiteness of the covariance matrix.

The RLS and RELS algorithms used in this work are implemented using the factorization algorithm of Bierman (1976). This method is based on the factorization of P into:

$$P = U\,D\,U^T \qquad (4.17)$$

where D is diagonal and U is an upper-triangular matrix. Åström and Wittenmark (1984) set out a Pascal routine for least squares estimation using the U-D factorization algorithm. For this work this code has been rewritten in FORTRAN.

4.3.4 Nonzero Mean Data

Most input and output signals seen in industrial control loops are nonzero mean and are related by a process model which includes a bias or offset as shown in Equations 3.4 and 3.6. In general, however, it is only the variations of the output signal with respect to the input signal which should be used in the parameter estimation algorithm (Isermann, 1980, 1982). The presence of nonzero mean input and output signal levels and a process bias complicates the recursive estimation of the process and noise model parameters. Various approaches to handle nonzero mean levels and process bias are suggested. Isermann (1980, 1982, 1986) summarizes three methods:

1. Use of incremental variables - If the first difference of the input and output signals is used instead of

their full value, the process bias, d, disappears (assuming d is constant) and the input-output regressor variables passed to the parameter estimation algorithm become zero mean, i.e.:

$$
\begin{aligned}
\Delta y(t) &= y(t) - y(t-1) \\
\Delta u(t) &= u(t) - u(t-1)
\end{aligned}
\qquad (4.18)
$$

Incremental variables arise naturally from the ARIMA system representation outlined in Section 3.1 and is the focus of this work. They do, however, have the disadvantage of being sensitive to high frequency noise so proper filtering must be used beforehand in the presence of this noise.

2. Use of deviation variables - This method removes the steady state levels from the input and output signals and defines a new deviation variable as:

$$
\begin{aligned}
y'(t) &= y(t) - y_{ss} \\
u'(t) &= u(t) - u_{ss}
\end{aligned}
\qquad (4.19)
$$

The mean deviation variables, u'(t) and y'(t), are then passed to the recursive algorithm for parameter estimation. One requirement of this method is the determination of the unknown steady state levels. This may be achieved by using a simple averaging filter. If required the process bias, d, is calculated as:

$$\hat{d} = (1 + \Sigma\hat{a}_i)\, y_{ss} - \Sigma\hat{b}_i\, u_{ss} \qquad (4.20)$$

3. Use of positional variables with a "1-in-the-data-vector" - The absolute signal values of u(t) and y(t) may be used directly in the input-output regressor of the parameter estimation algorithm if the process bias is estimated in addition to the dynamic parameters. The parameter and I/O vectors are augmented in the following way:

$$\theta(t) = [\hat{a}_1 \ldots \hat{a}_{n_a};\hat{b}_0 \ldots \hat{b}_{n_b};\hat{d}]^T \qquad (4.21)$$

$$\phi(t) = [-y(t-1) \ldots -y(t-n_a);$$
$$u(t-k) \ldots u(t-k-n_b);1]^T \qquad (4.22)$$

This scheme arises naturally from the ARMA system representation detailed in Section 3.1. A disadvantage of this scheme is the slow convergence of the bias estimate due to the fact that the "1-in-the-data-vector" is not a persistently exciting signal (Tuffs and Clarke, 1985). Another related problem is the interdependence of the dynamic parameter estimates and the bias estimate. Unbiased dynamic parameter estimates are not obtained until the bias estimate has converged to its true value.

4.3.5 Type of Identification

As mentioned in Chapter 3, self-tuning controllers can be divided into two broad categories, explicit/indirect and implicit/direct algorithms (Åström and Wittenmark, 1980; Isermann, 1986). This in turn results in explicit and implicit parameter estimation:

1. Explicit or indirect identification - In this arrangement the process model parameters are estimated and stored as intermediate results. The controller design calculations are then carried out using the estimated process parameters by invoking the certainty equivalence principle. Some advantages of this method are greater freedom of design, enabling modular programming, direct access to process parameter estimates and accommodation of variable time delays.
2. Implicit or direct identification - This method recasts the regression model of the process so that the controller parameters are estimated directly. The process parameters are contained implicitly and not as intermediate results. An advantage of this method is the reduced calculation time as the controller design calculations are eliminated. A limitation is that k must be known exactly.

Each of the self-tuning PID controllers developed in Chapter 3 has one formulation which is explicit. The focus of this work is this explicit or indirect formulation with the coefficients of $A(z^{-1})$, $B(z^{-1})$, and $C(z^{-1})$ process model polynomials being identified.

4.4 Model Reduction Technique

Reduced-order modelling is a technique whereby a high order process is represented by a lower order model which retains certain of the process properties such that model may be viewed as a reasonable approximation to the higher order process according to some specified criteria. This technique has been used in self-tuning control by de la Cruz et al. (1983) and Warwick (1985).

In the area of self-tuning or adaptive control a reduction in model order can be made at several stages (Warwick, 1985). The reduction can occur during the parameter estimation step by choosing the order of the estimated model to be lower than the original system order. This involves parameter estimation in the presence of process/model mismatch and can present potential problems if the mismatch is significant. Model reduction can also be made during the controller design stage. A sufficiently high order model of the process is estimated, and then the high order model is reduced using systematic model reduction techniques. This technique has the advantage that the user can specify the criterion that the reduced order model must meet (i.e. initial dynamics, steady state accuracy or a combination of both). Ashoor and Singh (1982) and Warwick (1984) give a review of available methods.

Of interest in this section is the second method - estimation of higher order process parameters followed by model order reduction by a systematic technique. The particular method used in this work is the approach proposed by Warwick (1984). It involves definition of a discrete error polynomial whose coefficients indicate the difference at any instant in time between the higher order process and the reduced order model. The error polynomial coefficients are determined by matching certain of the Markov parameters and certain of the time series proportionals (Padé approximation) between the reduced-order model and the process. The reduced-model coefficients are then calculated. A detailed development of this method can be found in Warwick (1984, 1985) and is summarized below.

Consider a general deterministic transfer function process of the form:

$$G(z^{-1}) = \frac{B(z^{-1})}{A(z^{-1})} \tag{4.23}$$

where $A(z^{-1})$ and $B(z^{-1})$ are polynomials in the backward shift operator z^{-1} represented by:

$$\begin{aligned} A(z^{-1}) &= 1 + a_1 z^{-1} + \dots + a_{n_a} z^{-n_a} \\ B(z^{-1}) &= \phantom{1 + {}} b_1 z^{-1} + \dots + b_{n_b} z^{-n_b} \end{aligned}$$

The desired reduced-order transfer function model can be described by:

$$R(z^{-1}) = \frac{C(z^{-1})}{D(z^{-1})} \tag{4.24}$$

where $C(z^{-1})$ and $D(z^{-1})$ are again polynomials in z^{-1}

represented by:

$$C(z^{-1}) = c_1 z^{-1} + \ldots + c_{n_c} z^{-n_c}$$
$$D(z^{-1}) = 1 + d_1 z^{-1} + \ldots + d_{n_d} z^{-n_d}$$

In addition both the process and the reduced-order model transfer functions can be written as power series expansions. The power series expansion about $z=\infty$ is an infinite series whose coefficients are called Markov parameters. The power series expansion about $z=1$ is an infinite series whose coefficients are called time series proportionals. The reduced order modelling problem is one of determining the parameters of $C(z^{-1})$ and $D(z^{-1})$ from the known polynomials $A(z^{-1})$ and $B(z^{-1})$ so that $R(z^{-1})$ is a good approximation to $G(z^{-1})$ according to some specified criteria.

An error polynomial is defined as:

$$W(z^{-1}) = B(z^{-1})D(z^{-1}) - A(z^{-1})C(z^{-1}) \qquad (4.25)$$

where

$$W(z^{-1}) = w_1 z^{-1} + w_2 z^{-2} + \ldots + w_{n_w} z^{-n_w} \qquad (4.26)$$
$$n_w = \max(n_b+n_d,\ n_a+n_c) \qquad (4.27)$$

The reduced-order model parameters can be determined by solving the system of equations resulting from Equation 4.25. The number of unknowns at this point is greater than the number of equations. The additional equations result from matching certain Markov parameters and certain time series proportional between the reduced-order model and the process. If the initial Markov parameters of the low-order model equal those of the process, the initial response of the model to a step input will approximate that of the system. In the same way, if the initial time series proportionals of the model and process are equated, the reduce-order model will approximate the process as the steady state is approached. The total number of points which can be matched is $2n_r$ where n_r is the order of the reduced order model. The matching of the first i Markov parameters of the reduced order model to the system can be done directly by equating the first w_i coefficients of $W(z^{-1})$ to zero. In order to match the time series proportionals of the model and the system, a modified form of the $W(z^{-1})$ polynomial is required. This $W'(z^{-1})$ polynomial is defined as:

$$W'(z^{-1}) = w_1'z^{-1} + w_2'z^{-2} + \ldots + w_{n_w}'z^{-n_w} \qquad (4.28)$$

where the coefficients of $W'(z^{-1})$ are obtained from the coefficients of $W(z^{-1})$ by means of Pascal's triangle.

In this work the model reduction technique is used to reduce an estimated second order process model to a first order model for PI control law calculations instead of estimating the parameters of a first order model directly. This approach is adopted to reduce the order mismatch between the process and the estimated model under which the parameter estimation algorithm must operate and to explore the merits of a systematic model reduction technique. A worked example is now considered.

Consider a second order process as shown in Equation 4.23 which is to be reduced to a first order model. The coefficients of the $W(z^{-1})$ error polynomial can be calculated from:

$$\begin{vmatrix} w_1 \\ w_2 \\ w_3 \end{vmatrix} = \begin{vmatrix} b_1 & 0 & -1 \\ b_2 & b_1 & -a_1 \\ 0 & b_2 & -a_2 \end{vmatrix} \begin{vmatrix} 1 \\ d_1 \\ c_1 \end{vmatrix} \qquad (4.29)$$

where a_i and b_i are the parameters of the second order system and c_i and d_i the first order model parameters. The above system of equations has 5 unknowns. It is desired to have the reduced order model match the second order system at the steady state. Matching the first two time series proportionals coefficients of the system and reduced-order model results in two more equations. Now the system can be solved. The $W'(z^{-1})$ polynomial coefficients are derived from the $W(z^{-1})$ polynomial using Pascal's triangle as:

$$\begin{aligned} w_1' &= w_1 \\ w_2' &= 2w_1 + w_2 = 0 \\ w_3' &= w_1 + w_2 + w_3 = 0 \end{aligned} \qquad (4.30)$$

With these extra equations, the relationships shown in Equation 4.29 is rearranged into the form:

$$\begin{vmatrix} -b_1 \\ -b_2 \\ 0 \end{vmatrix} = \begin{vmatrix} -1 & 0 & -1 \\ 2 & b_1 & -a_1 \\ -1 & b_2 & -a_2 \end{vmatrix} \begin{vmatrix} w_1 \\ d_1 \\ c_1 \end{vmatrix} \qquad (4.31)$$

This system of equations is solved for w_1, d_1 and c_1 given the parameters of the second order system. For computer implementation Equation 4.31 is rearranged so that c_1 and d_1 are calculated explicitly from the second order system parameters and are given by:

$$c_1 = \frac{(b_1+b_2)^2}{b_1(1-a_2) + b_2(2+a_1)} \tag{4.32}$$

$$d_1 = (b_1 + (a_2-1)c_1) / b_2 \tag{4.33}$$

4.5 Filtering

A practical difficulty encountered in the implementation of process control on real plants is the problem of noise. This noise associated with analog signals results from the process itself, the measurement device and the electrical and/or electronic equipment during ~~transmission~~ (Seborg et al., 1983). The process induced noise may arise from variations due to mixing, turbulence and non-uniform multiphase flow. The measurement noise comes from vibrations or instrument noise. This noise may be low or high frequency with high frequency noise introducing the problem of aliasing. During the sampling procedure of digital control systems, high frequency noise may be approximated as a low frequency component due to aliasing (Åström and Wittenmark, 1984).

Noise and its effects can be reduced by proper electrical practice and filtering of analog signals. The electrically generated noise can be minimized by shielding of cables, proper location of cables in conduit racks and grounding of equipment (Seborg et al., 1983). The other method of noise reduction is filtering of analog signals. Two types of filtering will be considered: analog filtering and digital filtering.

The simplest solution in handling the aliasing of high frequency noise is to introduce a low pass analog filter of the form:

$$G(s) = \frac{a}{s + a} \qquad (4.34)$$

between the process sensor and the sampler of analog to digital converter (ADC) of the digital control system. This prefilter will attenuate noise above the prefilter breakpoint "a". The particular breakpoint specified is also a function of the sample rate. If the sampling rate is changed, the prefilter may also require modification. One situation in process control where prefiltering can not be used is when a process stream is sampled and sent to an instrument for analysis (Åström and Wittenmark, 1984). Noise reduction in this case would involve taking a number of samples and mixing them before sending them to the process analyzer (i.e. on-line gas chromatograph). Further details on analog filtering are given in Seborg et al. (1983), Åström and Wittenmark (1984), Franklin and Powell (1981) and Isermann (1981).

The use of analog filters for low frequency noise becomes expensive. Isermann (1981) suggests that attenuation of low frequency noise be handled by digital methods. A simple low pass first order digital filter can have the form:

$$y_f(t) = (1-a)y_m(t) + ay_f(t-1) \tag{4.35}$$

where y_f is the filtered output, y_m the measured output and a the filter constant. Other types of digital filters are summarized in Isermann (1981) and Seborg et al. (1983). A special first order, high pass, filter is:

$$\Delta y(t) = y(t) - y(t-1) \tag{4.36}$$

where Δ is the differencing operator. The use of incremental variables in the parameter estimation algorithm as outlined in Section 4.3 is this type of filter.

In the implementation of self-tuning controllers both analog and digital filters have application. The analog input and output signals can be filtered by a low pass analog filter to attenuate high frequency noise if aliasing is a problem. A low pass digital filter (Equation 4.35) has application in filtering the process parameter estimates so that large fluctuations are eliminated before they are used for control law calculations.

4.6 Time Delay Compensation

The occurrence of time delays in the process industries is common and their presence complicates the implementation of process control techniques. The reasons for its prevalence is usually due to the nature of the process and the measurement instrumentation. Transportation of fluids over long distances is considered be the most common cause of time delays. In addition some measuring devices require long cycle times for sampling and analysis of process streams. A gas chromatograph is a typical example of such a device. Stephanopoulos (1984) suggests that time delays are sometimes introduced by final control elements which require some time to develop the actuating signal or by control implemented by a human operator who takes significant time to think and take proper control action.

The net effect of time delays is the degradation of closed-loop response if conventional feedback controllers are used. This is due to disturbances going undetected for a significant period of time, control action being taken on a measurement which described the process awhile ago, and control action not affecting the process output until some time in the future (Stephenopoulos, 1984). The result of these factors is that the stability margin of the closed-loop system is reduced (i.e. the crossover frequency and ultimate gain decreases). Therefore, to maintain the stability of the closed-loop system, the gain of the controller must be reduced. This leads to a sluggish closed-loop response.

One method proposed to improve closed-loop performance in the presence of time delay is a time delay compensation (TDC) technique. Such a method attempts to predict the undelayed output of the process and feedback this prediction to the controller. The best known technique is the Smith predictor (Smith, 1957). It is a model-based approach which divides the process model into two parts: the model dynamics portion and the time delay portion. Smith's approach in the discrete-time is summarized in Figure 4.1 where $G_p(z^{-1})$ is the actual process with time delay, $G_{pr}(z^{-1})$ is the estimated process model without time delay, $G_d(z^{-1})$ the estimated process time delay and $G_c(z^{-1})$ the feedback controller. The feedback signal is the predicted undelayed output plus the model prediction error. The effect of a perfect Smith predictor is to remove the process time delay from the closed-loop characteristic polynomial. The effectiveness of the Smith predictor is based on the accuracy of the model and the estimated time delay. Herein lies the weakness of the Smith predictor. Modelling errors due to changes in the process dynamics and time delay cause deterioration of the controller performance possibly to the point of instability.

The modelling problem and the simulation of time delays with analog hardware were against the widespread use of the Smith predictor in the 1950's and 1960's. With the advent of digital computers and adaptive control techniques, the

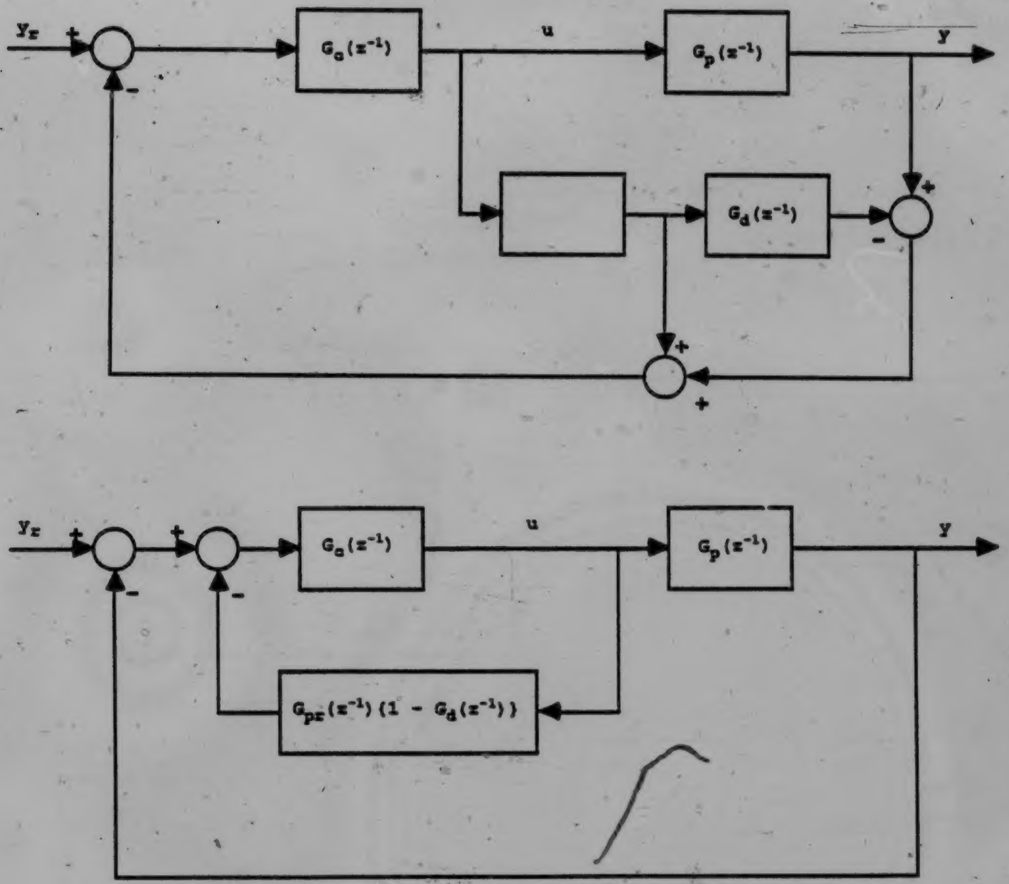
y_r
+
-
G_c(z^-1)
u
G_p(z^-1)
y
G_d(z^-1)
+
-
+
+
y_r
+
-
+
-
G_c(z^-1)
u
G_p(z^-1)
y
G_pr(z^-1)(1 - G_d(z^-1))

Figure 4.1 Equivalent block diagrams of the Smith predictor

feasibility of time delay compensation techniques like the Smith predictor has increased. If the process is time-invariant in terms of the dynamics and the time delay, off-line model identification can be used in conjunction with the TDC methods. If the structural parameters of a model, i.e. model order and time delay, are constant but the dynamic model parameters are time varying, standard recursive parameter estimation techniques can be used to implement the adaptive time delay compensator (Isermann, 1980, 1982, 1986). A more difficult problem is on-line search for model order and time delay as well as dynamic parameters. Schumnann et al. (1981) have proposed one method based on the information matrix P^{-1}. Vogel and Edgar (1980) note that many techniques proposed to handle this problem are unsatisfactory as a result of poor performance and/or excessive computational requirements. They conclude that the time delay is a difficult parameter to estimate on-line.

Some possible choices of time delay compensators from the literature are the adaptive Smith predictor (Habermayer and Kevicsky, 1985), the discrete analytical predictor (Doss and Moore, 1982), the generalized analytical predictor (Wellons and Edgar, 1985) and the adaptive time delay compensator of Vogel and Edgar (1980). The method chosen for this work was the Vogel and Edgar adaptive time delay compensator. It is a discrete TDC which is similiar in structure to the Smith predictor. *A priori* information required is the process model order and the upper and lower bound of the time delay. This second condition is less restrictive than the Smith predictor which requires exact knowledge of the time delay. The Vogel and Edgar adaptive TDC also handles processes with varying time delays.

The structure of this TDC is summarized in Figure 4.1 where $G_c(z^{-1})$ and $G_p(z^{-1})$ are as in the Smith predictor. $G_{pr}(z)$ and $G_d(z)$ are modified as:

$$G_{pr}(z^{-1}) = \frac{(\Sigma b_i)z^{-1}}{A(z^{-1})} \tag{4.37}$$

$$G_d(z^{-1}) = \frac{B(z^{-1})*z^{-k_{min}+1}}{\Sigma b_i} \tag{4.38}$$

where

$$\begin{aligned} A(z^{-1}) &= 1 + a_1 z^{-1} + \ldots + a_{n_a} z^{-n_a} \\ B(z^{-1}) &= b_0 + b_1 z^{-1} + \ldots + b_r z^{-r} \\ r &= k_{max} - k_{min} + n_b \end{aligned} \tag{4.39}$$

where n_a and n_b are the degree of the $A(z^{-1})$ and $B(z^{-1})$ polymonials respectively. The minimum and maximum time delay estimates of the process are denoted by k_{min} and k_{max}. Multiplication of $G_{pr}(z^{-1})$ and $G_d(z^{-1})$ result in an estimated process model $G_m(z^{-1})$ as shown:

$$G_m(z^{-1}) = G_{pr}(z^{-1}) * G_d(z^{-1}) = z^{-k_{min}} \frac{B(z^{-1})}{A(z^{-1})} \tag{4.40}$$

Vogel and Edgar (1980) contend that the difference between the true dynamic response and the dynamic response of the predicted output has a favourable effect on the closed-loop behavior. For a second order system with no continuous-time zeroes, the predicted model has faster dynamics. A root locus analysis of this shows that the discrete zero of the TDC process model has been moved to the origin of the unit circle.

4.7 Adaptive Feedforward Control

The basic idea behind feedforward control is to use the measurable load disturbance signals to anticipate the effect of the disturbance on the process output and to take appropriate compensating control action (Åström and Wittenmark, 1984). It is a very effective and well known way to reduce the influence of measurable disturbances. It has the advantage that action is taken before the disturbance is felt by the system and it does not introduce instability in the closed-loop response. Its disadvantage is that it requires a good dynamic process and disturbance model for systematic feedforward design. Another possible application of feedforward compensators is adaptive decoupling in multivariable system. Here the interaction is treated as a load disturbance and eliminated using feedforward compensation.

The model requirement problem can be overcome by the application of parameter adaptive control techniques (Schumann and Christ, 1979). The parameters of the process and load model can be obtained by a recursive parameter estimation method. Feedforward controller design can proceed according to the certainty equivalence principle. The only *a priori* information required is the sampling period and the structural parameters of the model - model orders and time delays. Various forms of the feedforward controller configuration are possible. Schumann and Christ (1979) consider seven feedforward control strategies. Only three methods, however, will be considered here in order show that adaptive feedforward compensation can be naturally included into the explicit self-controllers developed in Chapter 3. This work does not include simulation results of adaptive feedforward compensation.

Consider the following deterministic multi-input, single-output (MISO) system:

$$y(t) = z^{-k}\frac{B(z^{-1})}{A(z^{-1})}\,u(t) + z^{-l}\frac{L(z^{-1})}{A(z^{-1})}\,v(t) \qquad (4.41)$$

where $A(z^{-1})$, $B(z^{-1})$ and $L(z^{-1})$ are polynomials in the backward shift operator z^{-1}, and the time delays of the process and disturbance models are k and l respectively. The process input, measurable load disturbance and process output are u(t), v(t) and y(t) respectively. This MISO model can be derived from the ARIMA model of Equation 3.6 if the noise model and bias term is deleted and a measurable disturbance model added.

The design objective is to eliminate the effect of a measurable disturbance on the process output. This can be achieved if the feedforward compensator takes a completely dynamic form:

$$u_{ff}(t) = -z^{-(l-k)}\frac{L(z^{-1})}{B(z^{-1})}\, v(t) \qquad (4.42)$$

This dynamic feedforward compensator is realizable and applicable if $l \geq k$ and the polynomial $B(z^{-1})$ is stable (minimum phase).

A suitable approximation which can handle non-minimum phase processes is to design a partially static feedforward compensator of the form:

$$u_{ff}(t) = -z^{-(l-k)}\frac{L(z^{-1})}{\Sigma b_i}\, v(t) \qquad (4.43)$$

This avoids the instability problem caused by the nonmimimum phase $B(z^{-1})$ polynomial but the disturbance rejection capibility is compromised. A third alternative is a completely static feedforward controller of the form:

$$u_{ff}(t) = -z^{-(l-k)}\frac{\Sigma l_i}{\Sigma b_i}\, v(t) \qquad (4.44)$$

In general this compensator results in poor dynamic behavior of the controlled system but may be appropriate for deterministic step disturbances.

Implementation of each of the adaptive feedforward compensator listed in Equation 4.42 - 4.44 requires the online recursive estimation of the process and disturbance model parameters and a knowledge of the model orders and time delays. If the time delay estimates of the process and load models are difficult to obtain or the time delay is time varying (i.e. changing flow rates), an approach suggested by Vogel and Edgar (1980) may be used. This technique would extend the $B(z^{-1})$ and $L(z^{-1})$ polynomials to handle the uncertainty of the time delay of the process and disturbance models. For example, the partially static feedforward compensator would take the form:

$$u_{ff}(t) = -z^{-(l_{min}-k_{min})}\frac{L(z^{-1})}{\Sigma b_i}\, v(t) \qquad (4.45)$$

where

$$\begin{aligned} \Sigma b_i &= b_1 + b_2 + \ldots + b_r \\ L(z^{-1}) &= l_1 z^{-1} + l_2 z^{-2} + \ldots + l_s z^{-s} \\ r &= k_{max} - k_{min} + n_b \\ s &= l_{max} - l_{min} + n_l \end{aligned} \qquad (4.46)$$

and l_{min}, l_{max}, k_{min} and k_{max} are the minimum and maximum estimated time delays of the disturbance and process model respectively. The process and disturbance model orders are n_b and n_l respectively.

The use of this technique in conjunction with the completely dynamic FF compensator could lead to implementation problems since the straightforward application of Equation 4.42 in the presence of an overestimation of the process time delay will cause the first few b_i parameters to be zero or close to zero. This will result in a division by zero problem when the compensator control signal is calculated. This problem can be avoided by searching for leading zeroes and eliminating them by increasing the time delay estimate in the process model. The partially static and completely static FF compensators would also avoid this problem since the sum of the b_i parameters is used.

5. Linear SISO Process Models for Simulation Studies

5.1 Introduction

Chemical engineering industry processes are typically complex and nonlinear. They may be characterized as self-regulating or nonself-regulating, single-input single-output (SISO) or multiple-input multiple-output (MIMO), overdamped or underdamped systems. Most of the processes that control engineers must deal with are open-loop stable and overdamped and either SISO or MIMO. This does not rule out the occurrence or importance of nonself-regulating or underdamped processes.

Many of the today's advanced control techniques are model-based. Therefore, it is necessary to obtain a mathematical representation of the process which describes the physical and chemical phenomena occurring. Modelling of chemical processes is a discipline that requires all the principles of chemical engineering, such as thermodynamics, kinetics, transport phenomena etc. (Stephanopoulos, (1984). One approach is to obtain a linear input/output transfer function model of the process around a certain operating point. It has been said that most chemical processes can be adequately described by a first or at most a second order overdamped continuous transfer function cascaded with a time delay term of the form:

$$G(s) = \frac{K_p e^{-\theta s}}{(\tau_1 s + 1)(\tau_2 s + 1)} \qquad (5.1)$$

where K_p is the process gain, τ is the process time constant and θ is the time delay. An equivalent sampled-data representation can be obtained by cascading a zero order hold (ZOH) with Equation 5.1 and sampling at a rate T_s. The resultant second order discrete-time model is of the form:

$$G(z^{-1}) = \frac{(b_0 + b_1 z^{-1}) z^{-k}}{1 + a_1 z^{-1} + a_2 z^{-2}} \qquad (5.2)$$

where a_i and b_i are the sampled-data model parameters, k is the time delay of the process including the unit delay introduced by the ZOH and z^{-1} is the backward shift operator. Further details on the sampling of continuous-time plants are available in Franklin and Powell (1981), Phillips and Nagle (1984) and Stephanopoulos (1984). Equation 5.2 is in discrete polynomial form. It can be rearranged in pole-zero form by factoring the numerator and denominator polynomials. This results in a discrete model of the form:

$$G(z) = \frac{K_p'(z - z_1) z^{-k}}{(z - p_1)(z - p_2)} \qquad (5.3)$$

where z_1, p_1 and p_2 are the zeroes and poles respectively of the discrete process and K_p' is the discrete process gain.

The remainder of this chapter describes the linear SISO models used in the root locus and simulation studies of

Chapters 6, 7 and 8. The linear model structure will take on a more general form than those shown in Equations 5.1 - 5.3. Continuous process models of up to third order will be described as well as models with minimum phase zeroes and underdamped dynamics. The equivalent discrete process are described with the rationale for the sampling time choice.

5.2 Process Models

Table 5.1 summarizes the eight process transfer functions models used for stability and performance analysis in later chapters. Three forms of the process models are given: one in Laplace operator form and two in the sampled-data form. The range of possible sampling periods calculated by Isermann's criteria (Equation 4.1) is shown as well as the actual sampling period used. References are given if the process model has been extracted from the literature. There are three types of processes:

1. Overdamped process with no numerator dynamic in the continuous-time domain

   This category has four models ranging from first to third order. Process Model 2 (PM2) is the benchmark. This model has been used by Vogel (1982), Seborg, Shah and Edgar (1983), Tjokro and Shah (1985) and Chien et al. (1985). Process Model 3 is PM2 with lower frequency dynamics added. Process Model 4 is PM2 with higher frequency dynamics added. Both PM3 and PM4 have a

Table 5.1 Linear SISO process models for simulation studies

Model	Sampling Time	G(s)	G(z^{-1})	G(z)
PM1	$0.6<T_s<1.5$ $T_s=1.0$	$\frac{1}{(3s+1)}$	$\frac{.2835z^{-1}}{1-.7165z^{-1}}$	$\frac{.2835}{z-.7165}$
PM2	$1.3<T_s<3.2$ $T_s=1.0$	$\frac{1}{(3s+1)(5s+1)}$	$\frac{.028z^{-1}+.0234z^{-2}}{1-1.5353z^{-1}+0.5866z^{-2}}$	$\frac{0.028(z+.837)}{(z-.7165)(z-.8187)}$
PM3	$2.2<T_s<5.4$ $T_s=3.0$	$\frac{1}{(3s+1)(5s+1)(7s+1)}$	$\frac{.02614z^{-1}+.06379z^{-2}+.00948z^{-3}}{1-1.5681z^{-1}+.7991z^{-2}-.1315z^{-3}}$	$\frac{.02614(z+.159)(z+2.281)}{(z-.368)(z-.549)(z-.651)}$
PM4	$1.4<T_s<3.4$ $T_s=1.0$	$\frac{1}{(1s+1)(3s+1)(5s+1)}$	$\frac{.00768z^{-1}+.02123z^{-2}+.00357z^{-3}}{1-1.9031z^{-1}+1.151z^{-2}-.2158z^{-3}}$	$\frac{.00768(z+.180)(z+2.586)}{(z-.368)(z-.717)(z-.819)}$
PM5	$1.2<T_s<3.0$ $T_s=1.0$	$\frac{s+1}{(3s+1)(5s+1)}$	$\frac{.0791z^{-1}-.0277z^{-2}}{1-1.5353z^{-1}+0.5866z^{-2}}$	$\frac{0.0791(z-0.350)}{(z-.7165)(z-.8187)}$
PM6	$2.1<T_s<5.2$ $T_s=3.0$	$\frac{s+1}{(3s+1)(5s+1)(7s+1)}$	$\frac{.04809z^{-1}+.05300z^{-2}-.00168z^{-3}}{1-1.5681z^{-1}+.7991z^{-2}-.1315z^{-3}}$	$\frac{.04809(z-.031)(z+1.133)}{(z-.368)(z-.549)(z-.651)}$
PM7	$.21<T_s<.52$ $T_s=0.5$	$\frac{1}{s^2+1.5s+1}$	$\frac{.0969z^{-1}+.0754z^{-2}}{1-1.3001z^{-1}+0.4724z^{-2}}$	$\frac{0.0969(z+0.778)}{z^2-1.3001z+0.4724}$
PM8	$.21<T_s<.52$ $T_s=0.4$	$\frac{458}{(1s+1)(s^2+30s+229)}$	$\frac{.4673z^{-1}+.1889z^{-2}-.00090z^{-3}}{1-.6738z^{-1}+.0023z^{-2}-4.1E^{-6}z^{-3}}$	$\frac{.4673(z+.0048)(z+.399)}{(z-.67)(z^2-.0035z+6.E^{-6})}$

non-minimum phase zero in the discrete-time domain.

2. Overdamped process with one minimum phase zero in the continuous-time domain

   This category has two models. Process Model 5 is PM2 with one zero added at $s=-1$. Process Model 6 is PM3 with a zero added at $s=-1$. PM6 has a non-minimum phase zero in the discrete-time domain

3. Underdamped process with no numerator dynamics in the continuous-time domain

   This category has two models. Process Model 7 is second order and has been used by Zafiriou and Morari (1985). Process Model 8 is a third order model used by Rohrs et al. (1984) and Cluett et al. (1986).

6. Stability and Performance - A Root Locus and Time-Domain Analysis

6.1 Introduction

Three self-tuning PID controllers were derived in Chapter 3. The self-tuning robust controller (SRC) and the integrating STC based PID controllers, derived from a generalized minimum variance (GMV) design strategy, can be shown to be identical for the explicit velocity "setpoint-on-P&I&D" formulation. The third PID controller results from a pole placement criteria and can also be formulated in the explicit velocity "setpoint-on-P&I&D structure. The object of this chapter is to compare these two types of controllers on some linear SISO process models in order to form some conclusions as to their characteristics.

The basis of comparison is two-fold : stability analysis focusing on the root locus method and performance analysis from time-domain responses. Stability consideration are of prime importance for any closed-loop control system. The controller must be designed so that the closed-loop poles are in a stable region : the left half of the s-plane or the interior of the unit circle in the z-plane. To achieve stability of the closed-loop system, the designer may modify the gain, poles and zeroes of the controller assuming that the process is fixed. In a velocity PID structure, the controller poles are fixed. Only the controller gain and zeroes can be modified by varying the controller gain, integral and derivatives times. It is also required that the closed-loop poles of the control system be located so that desirable time-domain characteristics be achieved such as transient behaviour and steady state response.

Some stability analysis techniques available for continuous-time systems are the Routh-Hurwitz criterion, the Nyquist and Bode frequency response methods and the root locus method. Some of these techniques can be applied to discrete-time systems via a bilinear transformation whereas the root locus method may be applied directly to discrete-time systems. It is the root locus method that is used to compare the generalized minimum variance and pole placement self-tuning PID controllers for stability. Insights into the controller time-domain performance, sensitivity to process/model mismatch and parameter tuning also result from this analysis.

The second method for comparison of these two controllers types is the time-domain response of the system to a unit step change in the system setpoint. The basis for comparison is the observed rise time (t_r) and settling time (t_s) for a fixed actual peak overshoot (M_p). The controller gain (K_c) required for the given response is also considered.

This chapter will compare the stability and performance of the generalized minimum variance (GMV) and pole placement (PP) based PID controllers on eight linear, time invariant

SISO transfer function models with varying structure and order and with varying degrees of process/model mismatch. Section 6.2 presents a brief summary of the root locus technique. Section 6.3 documents a typical data set of the controllers with a particular model. Section 6.4 compares the GMV AND PP PI controllers for the eight process models. Section 6.5 compares the PID controllers. Section 6.6 discusses the results and draws some general conclusions.

6.2 The Root Locus Technique

A root locus curve is simply a complex plane plot of the roots of the characteristic polynomial of a closed-loop control system as the controller gain (K_c) is varied from zero to infinity. It is a graphical display of all the closed-loop poles as a function of K_c and may be used to determine the stability and performance characteristics of a closed-loop system.

A root locus plot may be constructed by solving the characteristic polynomial explicitly as K_c is varied and plotting the results or by qualitatively sketching the root locus by following a set of rules. The first method is increasingly more common with the availability of computers. The second method is generally covered in most standard control texts (Ogata, 1970; Dorf, 1980; Phillips and Nagle, 1984). Phillips and Nagle (1984) list the following rules for the construction of a root locus plot :

1. The root loci originate at the open-loop poles of the process and controller and terminate at the open-loop zeroes of the process and controller.
2. The root locus on the real axis is determined by the location of real open-loop poles and zeroes and always lies to the left of an odd number of real open-loop poles and zeroes. Complex-conjugate open-loop poles and zeroes have no effect on the location of the root locus on the real axis.
3. The root loci are symmetrical with respect to the real axis of the complex plane.
4. Breakaway points and break-in points either lie on the real axis or occur in complex-conjugate pairs. They are given by the roots of :

$$\frac{d}{dz}\{1 / [D(z)G(z)]\}$$

   where D(z) is the transfer function of the controller and G(z) the transfer function of the process.

The rules for constructing a root locus plot for discrete-time systems are the same as those for continuous-time systems. However, the interpretation of the root loci plot with respect to stability and dynamic response is different. In the complex s-plane the stable region is the left half plane. The stable region in the z-plane is the interior of the unit circle.

The root locus analysis of this work is in the discrete-time domain. Various controller and process configurations are analyzed for stability boundaries by plotting the root locus curve and calculating the K_c which causes the root loci to cross the unit circle. As mentioned earlier root locus plots not only provide stability information but also give information about the general dynamic response of the closed-loop system as a function of K_c and controller zero location. These same controller and process configurations are analyzed qualitatively for dynamic responses from the root locus plot because the basic characteristics of the transient response of the closed-loop system is determined by the closed-loop poles locations.

6.3 Method of Comparison - A Typical Data Set

An analysis is presented comparing the GMVPI(D) and PPPI(D) controllers for the eight process models documented in Chapter 5. The complete set of root locus plots are included in this chapter but only a selected number of time-domain responses are presented. Those presented are typical for the ideal and model-plant mismatch cases. To illustrate the method used for this comparison, a detailed data set is presented for a ideal case along with the basis of the comparison.

Process Model 2 (PM2) is an overdamped, second order process with no numerator dynamics (see Table 5.1). The discrete transfer function for T_s=1 is of the form :

$$G_2(z) = \frac{0.0280z + 0.0234}{z^2 + 1.5353z + 0.5866} \tag{6.1}$$

For an ideal case the process order (P), estimated model order (M) and the controller order (C) must be equal. For this example a PID controller (C=2) is used. The structure of the controller is always the "setpoint-on-P&I&D" form. Both the GMV and PP based PID controllers are designed from the process parameter estimates which are assumed to have converged to a "reasonable" set of values. If there is no mismatch between process order and controller order, as is the case in this example, the true parameter values are used (i.e. M=2). The GMVPID controller calculates the integral time and derivative time as:

$$T_i = -(a_1 + a_2)\,T_s = 0.949$$

$$T_d = \frac{-a_2\,T_s}{(a_1 + a_2)} = 0.618$$

These constants fix the controller zeroes at 0.418±j0.237. The PPPID controller calculates T_i and T_d as:

$$T_i = \frac{-(a_1 + 2a_2)T_s}{1 + a_1 + a_2} = 7.058$$

$$T_d = \frac{-a_2\,T_s}{(a_1 + 2a_2)} = 1.620$$

This results in the controller zero locations of z = 0.7165 and z = 0.8187, exactly cancelling the process poles.

The root locus plot of PM2 with the GMVPID and PPPID controllers is shown in Figure 6.1 as the controller gain is varied from zero to infinity. The open-loop poles are marked with an "X". The PID controller contributes real poles at z=0 and z=1. The open-loop zeroes are marked with a "O". The closed-loop poles are marked with a "□" and correspond to a K_c which gives an actual peak overshoot of 15% for the closed-loop system. The value of K_c for these points is shown on the appropriate table. The real closed-loop poles are not generally shown as they often obscure the open-loop poles and zeroes. The root loci are the bold lines which are symmetric with the real axis. The stability boundary is shown as the unit circle. The controller gain is shown for breakaway and break-in points of the root loci. The controller gain at the stability boundary for the GMVPID controller is $K_c = 1.16$ and the PPPID controller is $K_c = 15.5$. A qualitative analysis of the root locus plots shows that the GMVPID controller has 4 branches and the complex portion of the root locus is very near the unit circle. The PPPID controller root loci has only 2 branches and the complex portion of the root locus is well within the unit circle.

The time-domain response of Process PM2 with the GMVPID and PPPID controllers for a unit step change in setpoint is shown in Figure 6.2. For each closed-loop system the actual time-domain peak overshoot is fixed as M_p=15%. Controller performance is then evaluated by comparison of the rise time and settling time for each closed-loop system. The rise

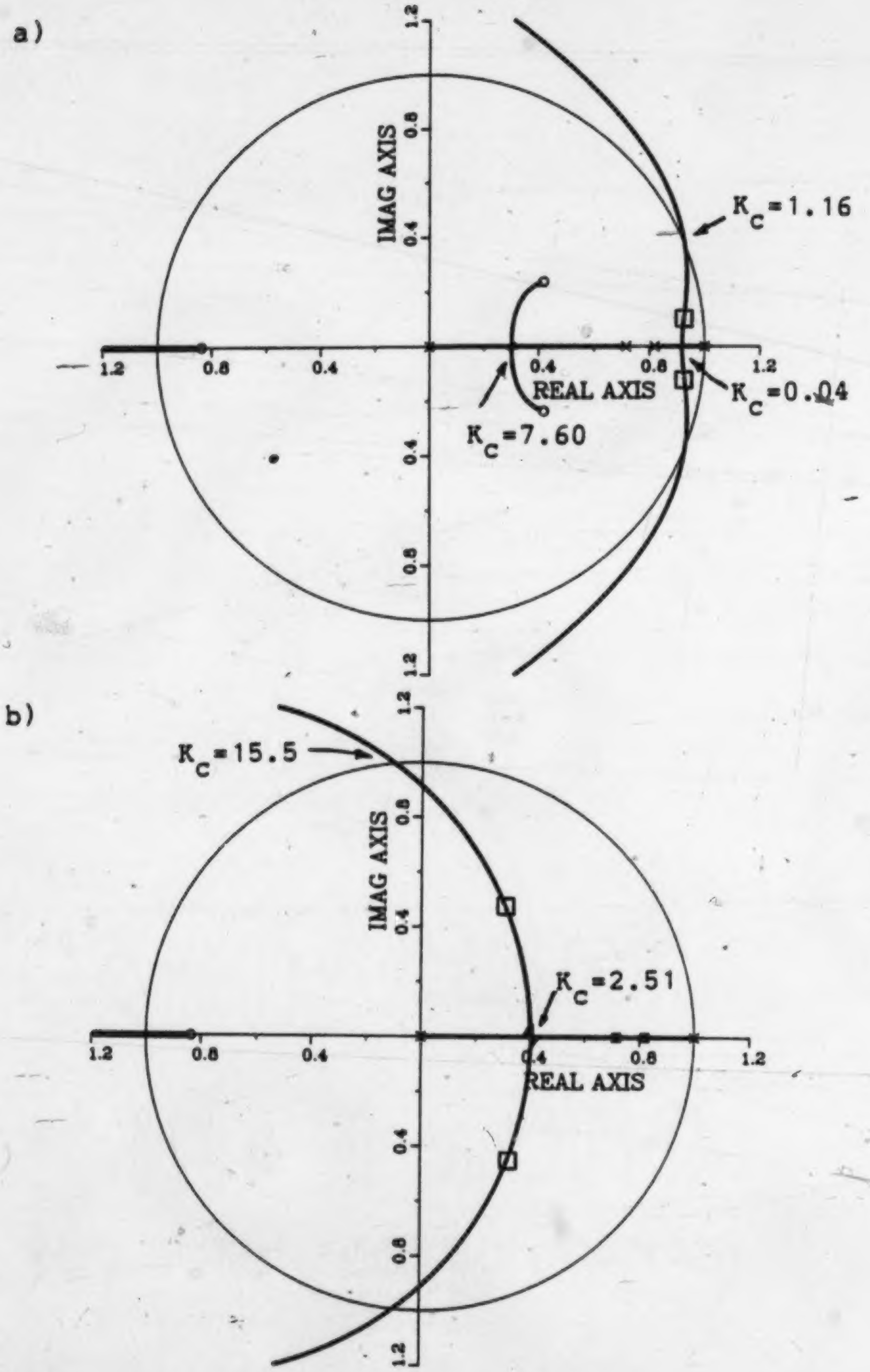

Figure 6.1 Root loci of Process Model 2 with a) GMVPID and b) PPPID controllers (P=M=C=2)

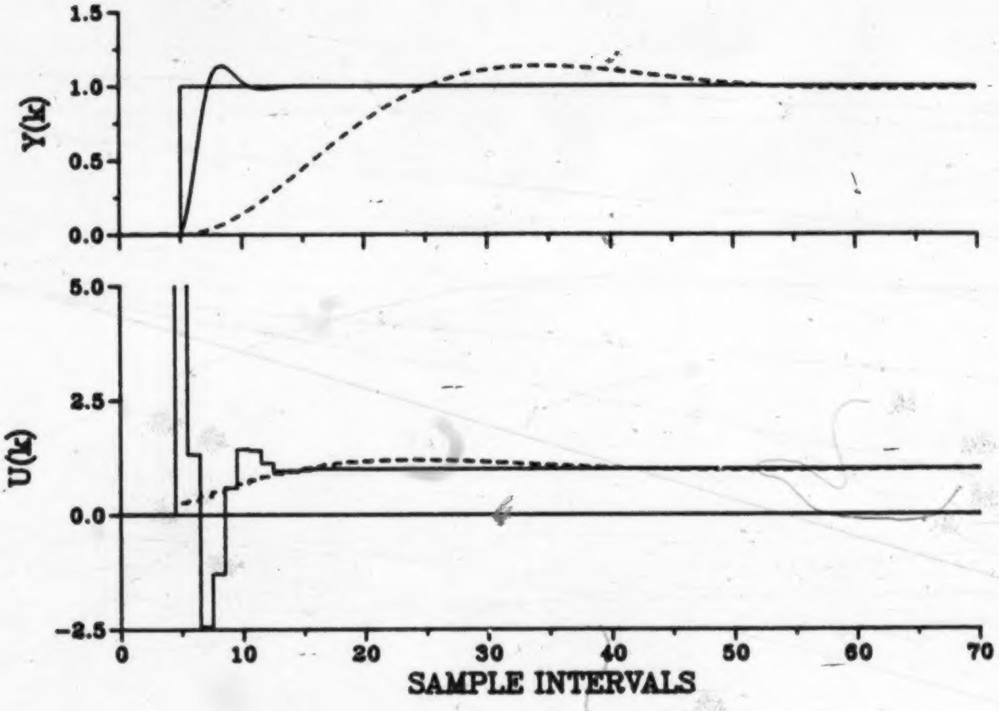

Figure 6.2 Servo response of Process Model 2 with GMVPID (----) and PPPID (——) controllers (P=M=C=2)

time is defined as the time required for the output to rise to within 5% of the new setpoint. The settling time is defined as the time required for the process outpoint to remain within a 1% deadband of the new setpoint. For the GMVPID control law the t_r and t_s are calculated to be 18.8 and 66.0 respectively. The PPPID algorithm resulted in a t_r=2.2 and t_s=7.8. The control effort of the PPPID control law is considerably greater than the GMVPID control law.

The results of the root locus analysis and the time domain analysis are summarized for comparison in tabular form. The ultimate controller gain, the rise time and the settling time are the criteria used to determine which of the two PID control laws has better characteristics. The individual criterion which is superior is marked with an asterisk. This comparison is completed for the eight process models for both the PI case and the PID case.

6.4 Comparison of GMV and PP Based PID Controllers

The results of the comparative study between the GMVPI and PPPI controllers are summarized in Table 6.1 for seven of the linear SISO models outlined in Chapter 5. The basis for determining which PI controller is superior is the ultimate controller gain (K_c^u) from the root locus plots and the rise time (t_r) and settling time (t_s) for an actual peak overshoot of M_p=15% from the time-domain response data. Table 6.1 also includes the integral time calculated by the GMVPI or PPPI controllers and the controller gain

Table 6.1 Comparison of GMV and PP based PI control laws

Process Model	Mismatch	Figure No.	GMVPI					PPPI				
			K_c^u K_c	T_i	t_r	t_s	Notes	K_c^u K_c	T_i	t_r	t_s	Notes
PM1	P=1	6.3	3.57	0.72	3.8	17.7	4	5.05	2.53	0.8	2.5	1
	M=1	6.4	0.41					2.90				
	C=1							*		*	*	
PM2	P=2	6.5	0.80	0.88	19.1	66.4	2	12.70	7.54	5.7	22.8	2
	M=2		0.096					2.10				
	C=1							*		*	*	
PM3	P=3	6.6	0.55	2.32	32.3	113.0	2	2.87	10.20	17.2	58.6	2
	M=2	6.7	0.13					0.93				
	C=1							*		*	*	
PM4	P=3	6.8	0.39	0.82	21.1	73.9	2	3.79	4.57	11.1	39.6	2
	M=2		0.017					0.77				
	C=1							*		*	*	
PM5	P=2	6.9	18.60	0.87	15.6	57.5	4	27.20	6.52	2.9	14.9	4
	M=2		0.12					4.00				
	C=1							*		*	*	
PM6	P=3	6.10	0.80	2.32	29.6	105.0	2	3.90	10.20	14.8	52.6	2
	M=2		0.15					1.08				
	C=1							*		*	*	
PM7	P=2	6.11	0.82	0.36	3.4	11.3	2	3.49	1.25	2.0	11.6	2
	M=2		0.18					0.88				
	C=1					*		*		*		

Notes: 1. no branches in root locus
2. two branches in root locus
3. four branches in root locus
4. circle pattern in root locus
5. complex controller zeroes

which produced the time-domain response with M_p=15%. The degree of process/model mismatch for each process is listed. Table 6.1 cross-references to the figure number for each process/controller combination. The root locus plots and transient runs are shown in Figures 6.3 - 6.11. These plots have been included to validate the analysis and provide a basis for discussion later. Finally, Table 6.1 qualitatively describes the nature of the root locus plots for the various process/controller combinations in summary form. Each root locus plot includes open-loop poles and zeroes, the closed-loop poles corresponding to the controller gain (Kc) listed on Table 6.1, which do not obscure open-loop poles and zeroes, and the breakaway/break-in points.

Table 6.1 clearly shows which PI controller is superior based on the three-fold criteria. Of the 21 criteria 20 asterisks are marked on the PPPI controller. Only the settling time of GMVPI for Process Model 7 is slightly better than the settling time of the PPPI controller. In most other cases the K_c^u, t_r and t_s of the PPPI controller are a factor of two better than the same parameters for the GMVPI controller. The general pattern of the root loci for each process model is the same. The main difference between the two controllers is the position of the controller open-loop zero which is located by the integral time. The T_i for the GMVPI controller is calculated simply as the estimated a_1 parameter multiplied by the sampling period. For the

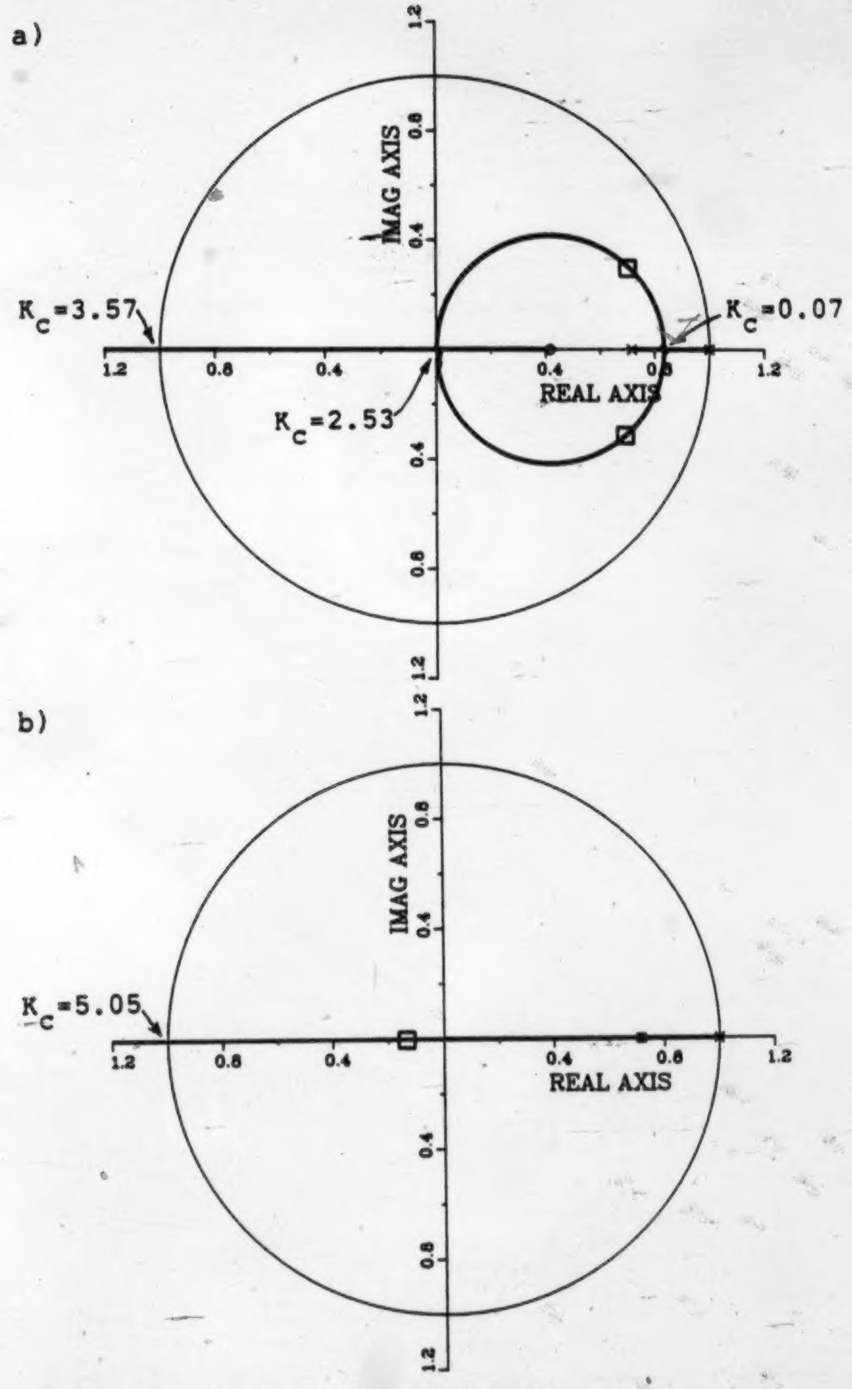

Figure 6.3 Root loci of Process Model 1 with a) GMVPI and b) PPPI controllers (P=M=C=1)

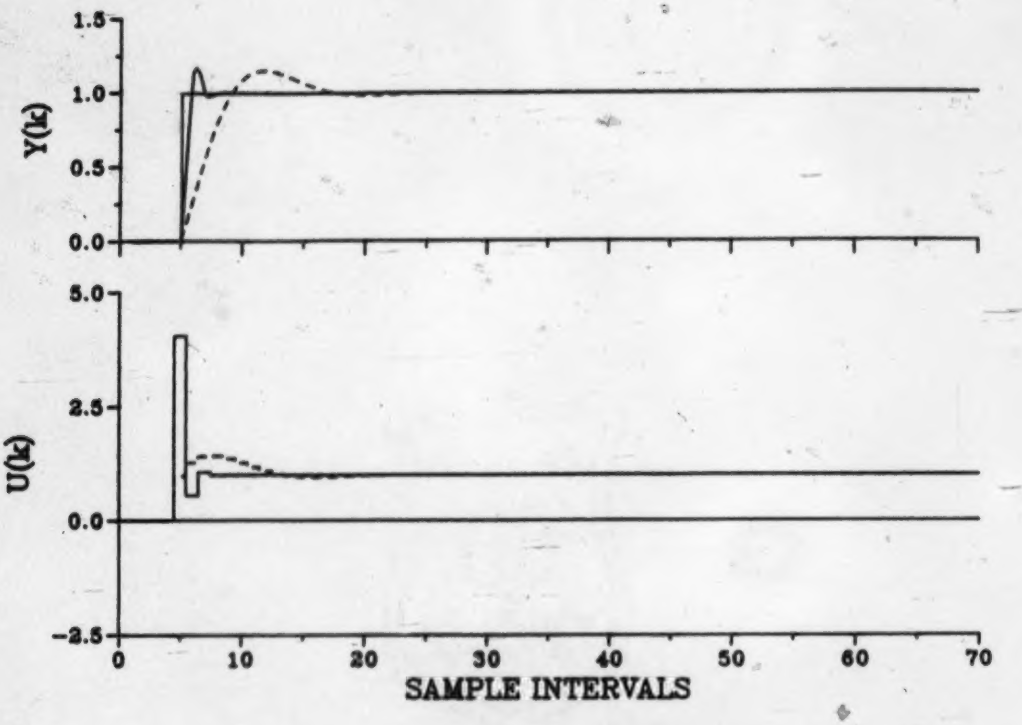

Figure 6.4 Servo response of Process Model 1 with GMVPI (----) and PPPI (——) controllers (P=M=C=1)

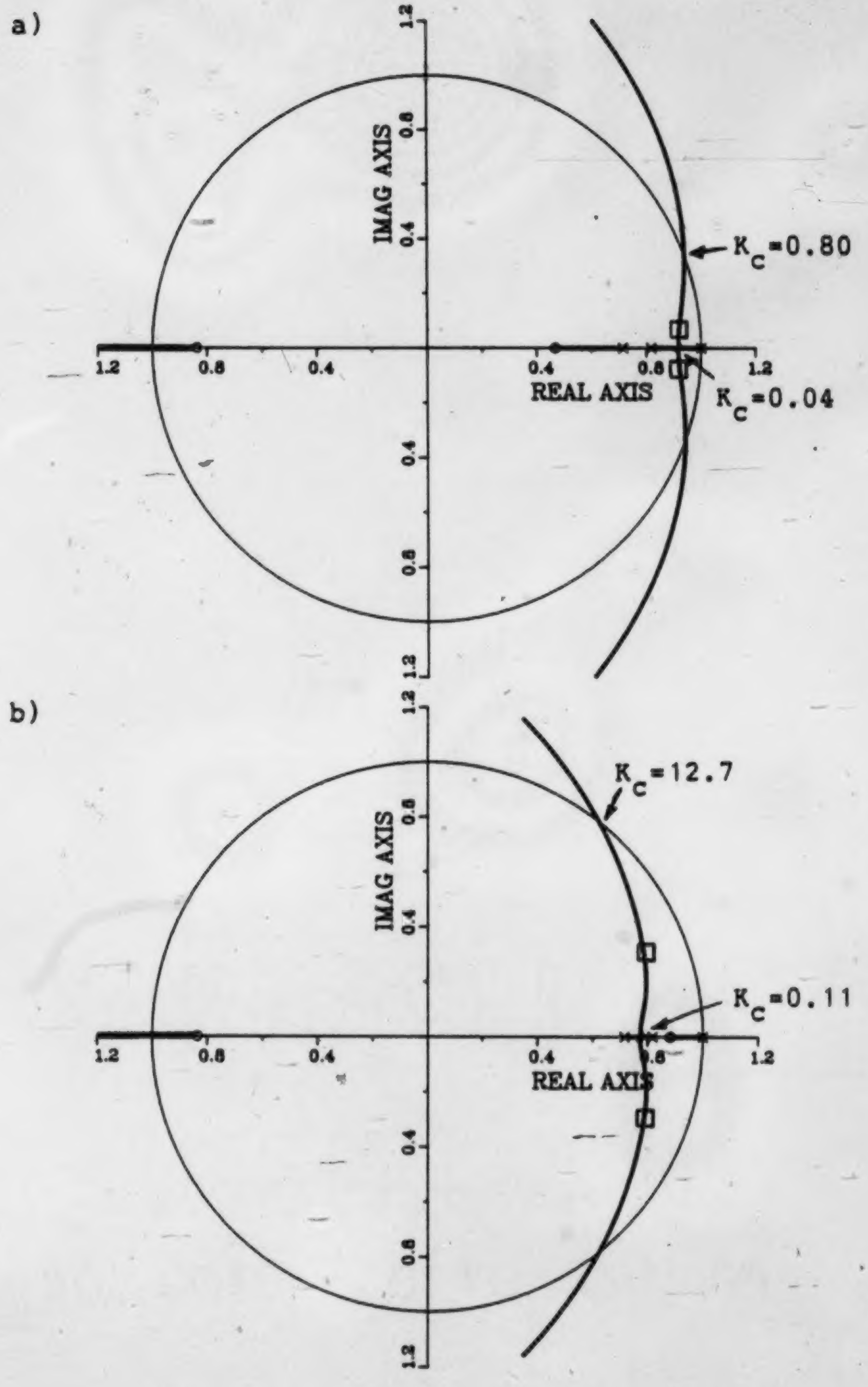

Figure 6.5 Root loci of Process Model 2 with a) GMVPI and b) PPPI controllers (P=M=2,C=1)

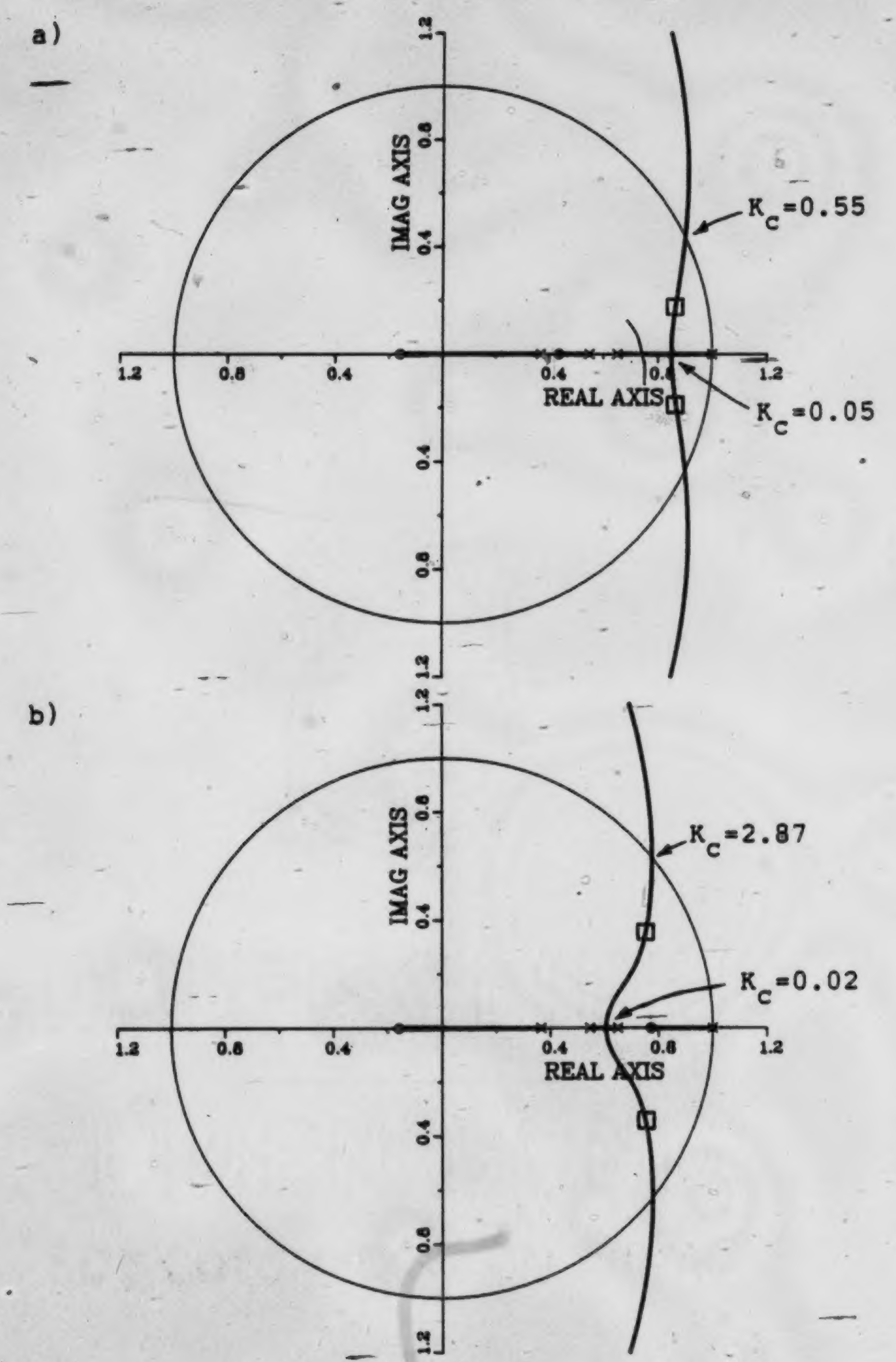

Figure 6.6 Root loci of Process Model 3 with a) GMVPI and b) PPPI controllers (P=3,M=2,C=1)

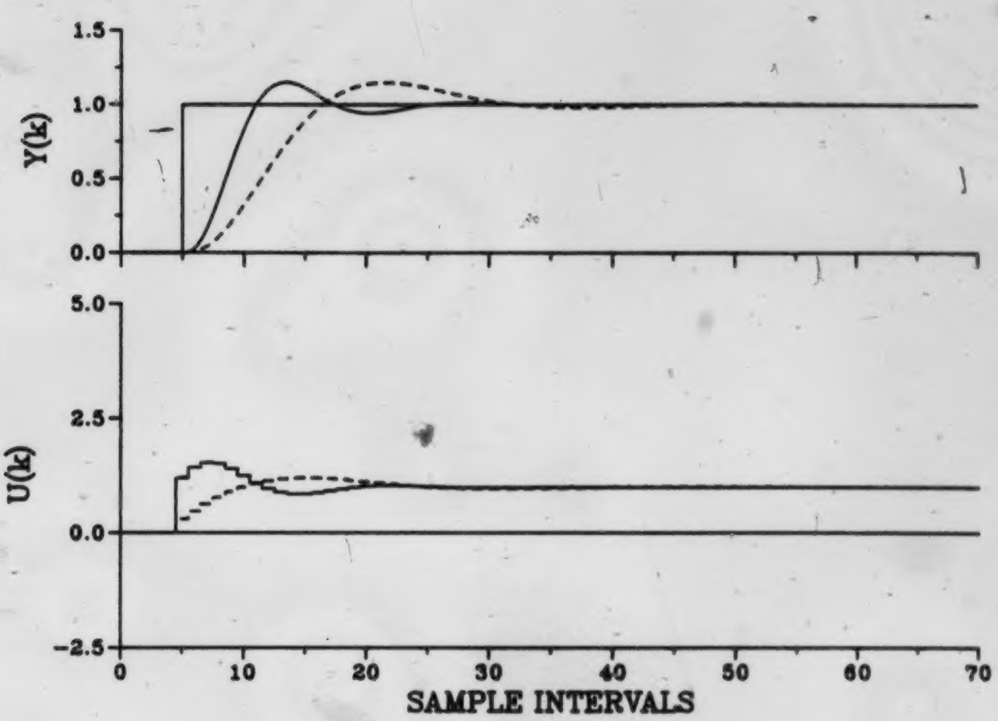

Figure 6.7 Servo response of Process Model 3 with GMVPI (----) and PPPI (——) controllers (P=3,M=2,C=1)

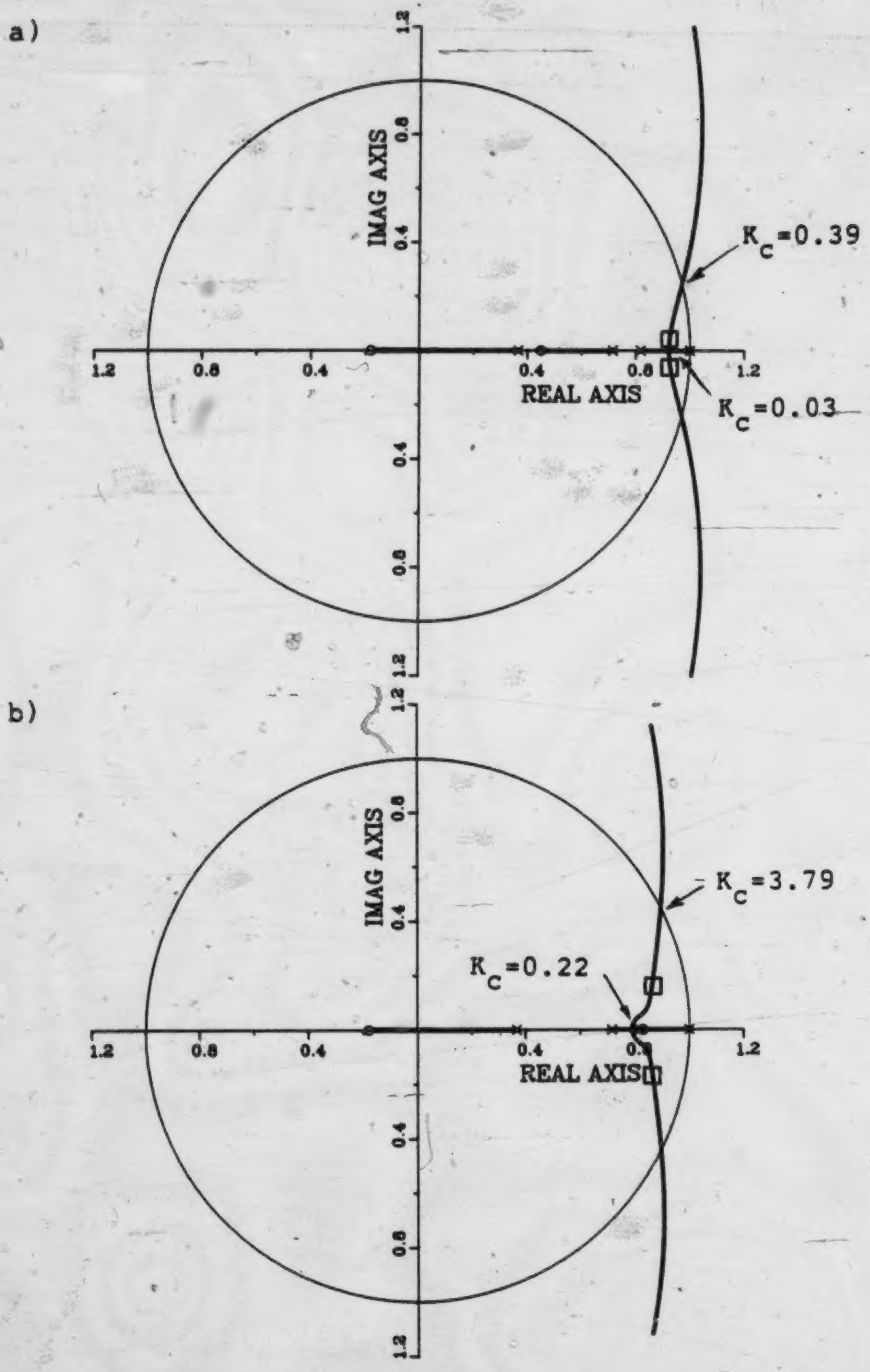

Figure 6.8 Root loci of Process Model 4 with a) GMVPI and b) PPPI controllers (P=3,M=2,C=1)

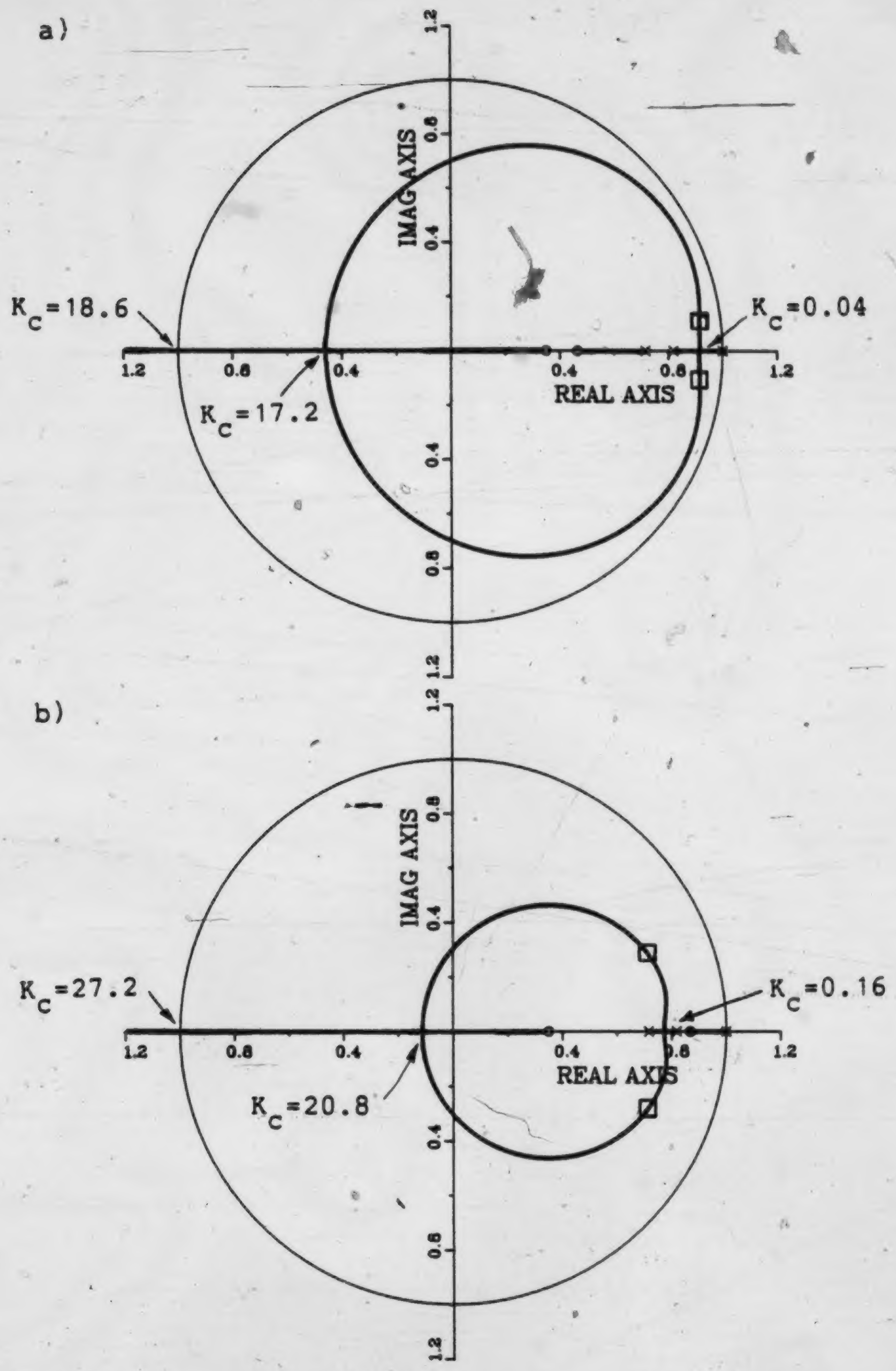

Figure 6.9 Root loci of Process Model 5 with a) GMVPI and b) PPPI controllers (P=M=2,C=1)

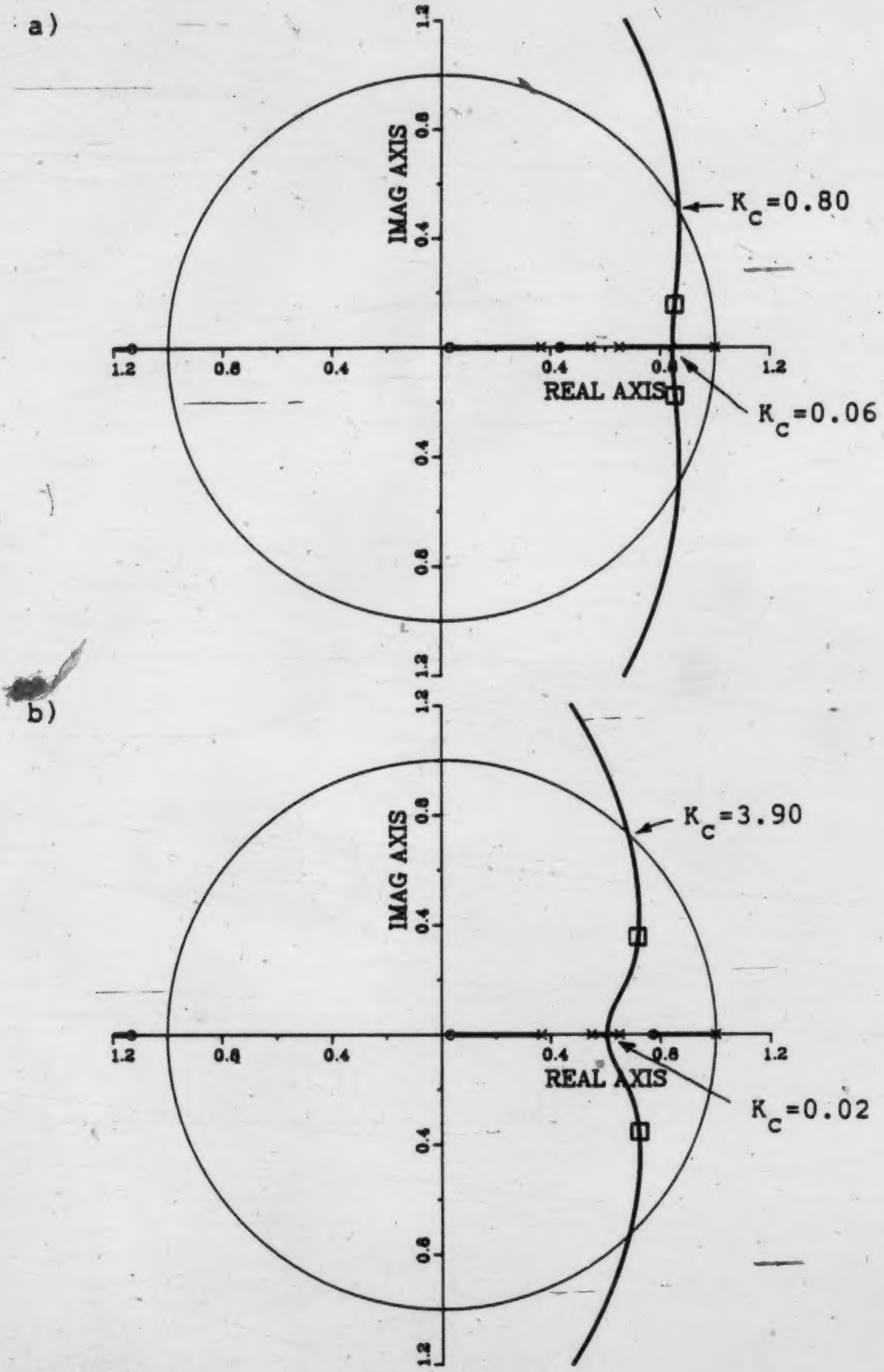

Figure 6.10 Root loci of Process Model 6 with a) GMVPI and b) PPPI controllers (P=3,M=2,C=1)

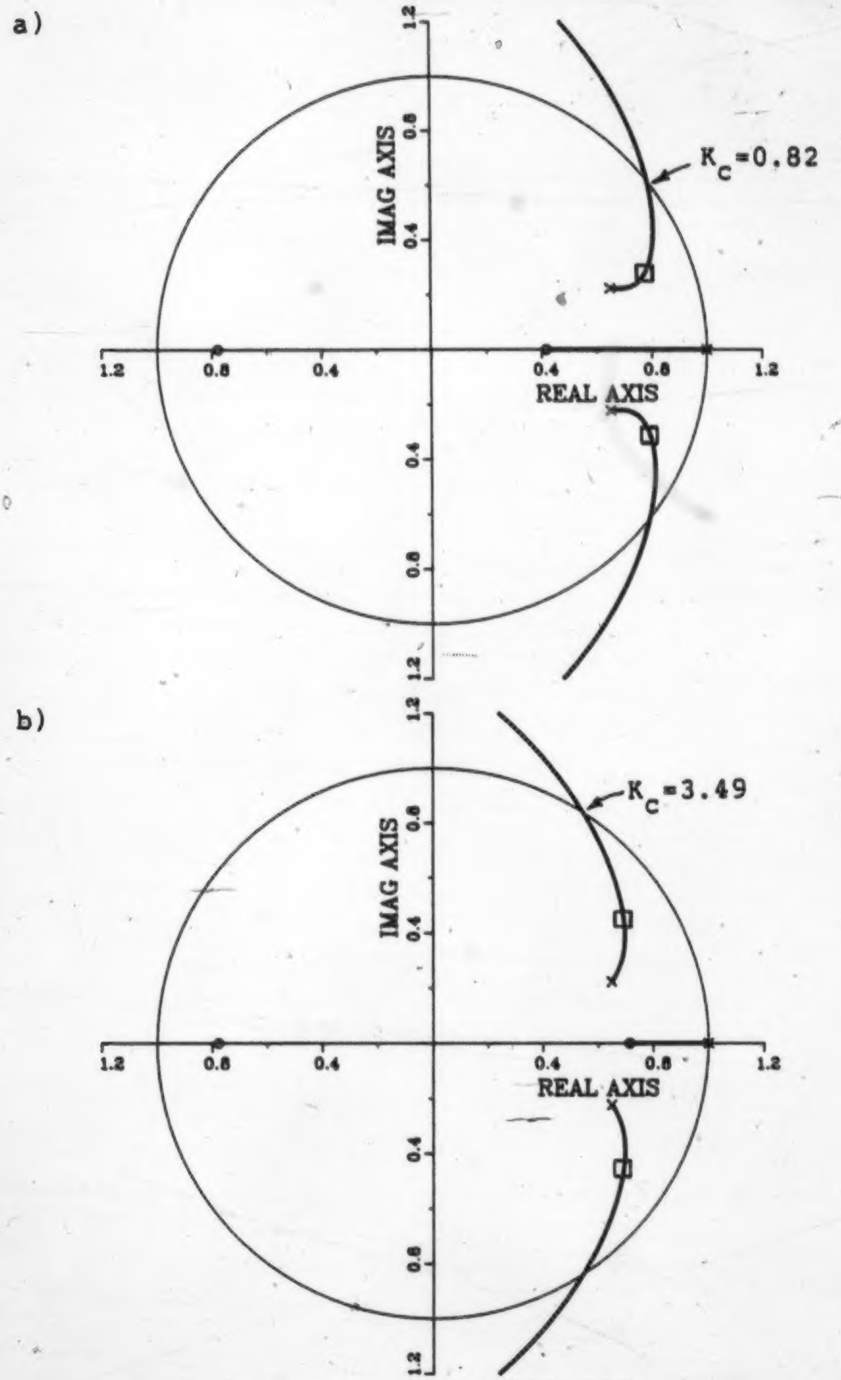

Figure 6.11 Root loci of Process Model 7 with a) GMVPI and b) PPPI controllers (P=M=2,C=1)

processes considered this generally placed the controller zero near z=0.4 on the real axis. This controller zero location causes branching between the controller pole at z=1 and the process pole nearest z=1. This branching occurred at very low controller gains (K_c<0.06). The complex portion of the root loci for the GMVPI controller is generally near z=1 close to the unit circle. The result of this location is the long rise and settling times.

The PPPI controller located its controller zero near the open-loop process poles. For the no mismatch case of PM1 this controller cancels the process pole. This location generally cases the branching to occur between the process poles at higher controller gains. The root loci are located well within the unit circle and results in a better t_r and t_s. The superiority of the PPPI controller over the GMVPI controller is illustrated for the ideal and mismatch case by the dynamic responses shown in Figures 6.4 and 6.7.

6.5 Comparison of GMV and PP Based PID Controllers

Results of the comparison between the GMVPID and PPPID controllers are summarized in Table 6.2 for six of the linear SISO models. The criteria for judging the better PID controller is the ultimate controller gain (K_c^u) from a root locus analysis and the rise time (t_r) and settling time (t_s) from the time-domain response data for an actual peak overshoot of M_p=15%. Table 6.2 includes the integral and derivative times calculated by the GMVPID and PPPID

Table 6.2 Comparison of GMV and PP based PID control laws

			GMVPID					PPPID				
Process Model	Mismatch	Figure No.	K_c^u K_c	T_i T_d	t_r	t_s	Notes	K_c^u K_c	T_i T_d	t_r	t_s	Notes
PM2	P=2	6.1	1.16	0.95	18.8	66.0	3,5	15.50	7.06	2.2	7.8	2
	M=2	6.2	0.10	0.62				4.70	1.62			
	C=2							*		*	*	
PM3	P=3	6.12	0.97	2.70	31.3	111.0	3,5	3.24	8.10	8.5	53.9	3,5
	M=2	6.13	0.16	2.09				1.46	6.92			
	C=2							*		*	*	
PM4	P=3	6.14	0.52	0.96	20.9	73.3	3,5	5.58	3.84	3.2	36.4	3,5
	M=2		0.089	0.82				2.30	4.52			
	C=2							*		*	*	
PM5	P=2	6.15	10.60	0.95	15.1	56.7	2,4,5	6.78	7.06	1.0	10.0	1
	M=2		0.13	0.62				5.25	1.62			
	C=2		*								*	*
PM6	P=3	6.16	3.90	2.65	28.7	104.0	3,5	5.05	8.54	7.3	37.7	3,5
	M=2		0.18	1.88				1.90	5.04			
	C=2							*		*	*	
PM7	P=2	6.17	1.95	0.41	3.1	10.9	3,5	4.71	1.03	1.0	3.8	2,5
	M=2		0.24	0.29				1.45	0.67			
	C=2							*		*	*	

Notes: 1. no branches in root locus
2. two branches in root locus
3. four branches in root locus
4. circle pattern in root locus
5. complex controller zeroes

controllers and the controller gain which produced the time-domain response of M_p=15%. The degree of process/model mismatch for each process is listed. Table 6.2 cross-references to the figure number for each process/controller combination. These root locus plots and transient run as shown in Figures 6.1, 6.2 and 6.12-6.17. These plots have been included to validate the analysis and facilitate later discussion. Finally, Table 6.2 qualitatively describes the nature of the root focus plot for each process/controller combination in summary form. Each root locus plot includes the open-loop poles and zeroes, the closed-loop poles corresponding to the controller gain (Kc) listed in Table 6.2, which do not obscure open-loop poles and zeroes, and the breakaway/break-in points.

Table 6.2 clearly shows that superiority of the PPPID controller over the GMVPID controller. For 17 out of the 18 criteria, the PPPID controller is judged to have the better characteristics. Only the $K_c{}^u$ of the GMVPID controller for PM3 is better. As is the case with the PPPI controller the $K_c{}^u$, t_r and t_s for the PPPID is generally a factor of two better than those of the GMVPID controller.

An analysis of the root locus plots shows that the GMVPID controller places complex controller zeroes at approximately z=0.4±j0.2. The complex controller zeroes are caused by $T_d>T_i/4$. As is the case with the GMVPI controller, the location of these zeroes generally cause 4 branches in the root loci with the breakaway points at low

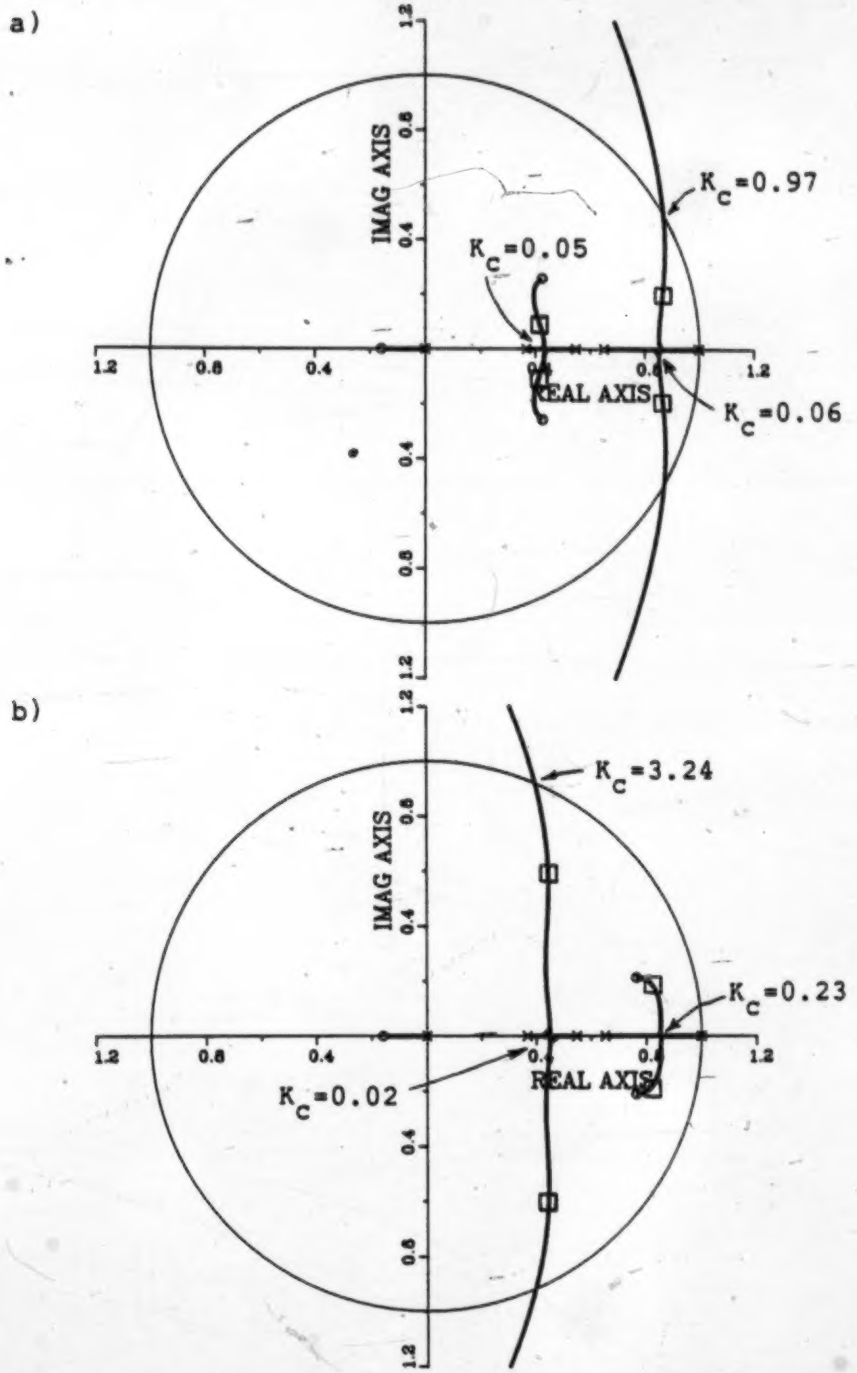

Figure 6.12 Root loci of Process Model 3 with a) GMVPID and b) PPPID controllers (P=3,M=C=2)

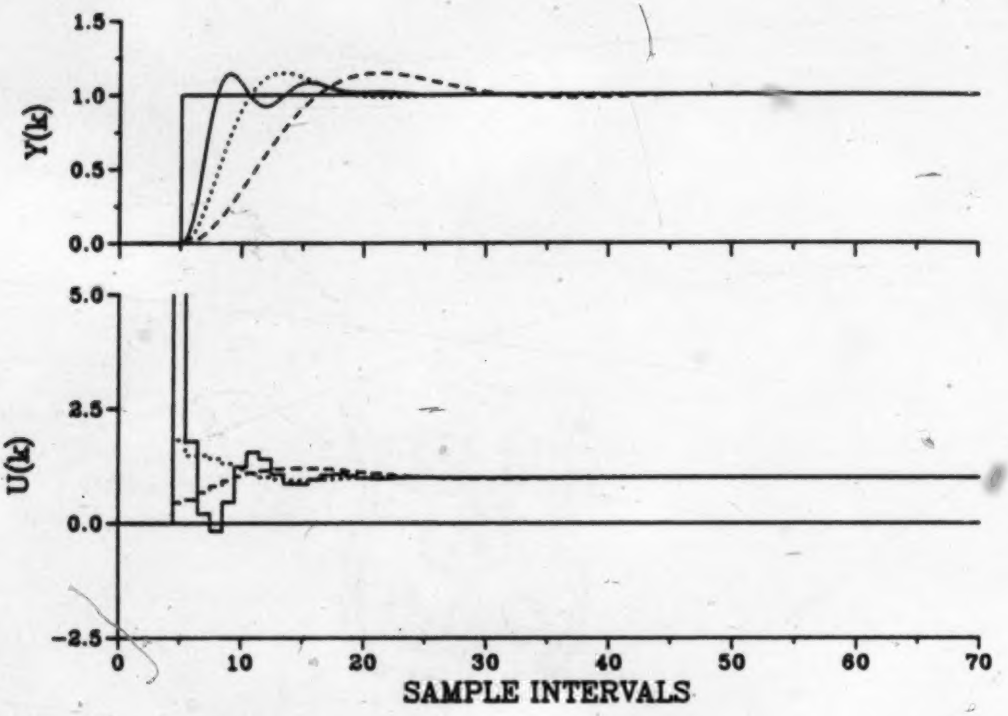

Figure 6.13 Servo response of Process Model 3 with GMVPID (----), PPPID (——) and modified PPPID (····) controllers (P=3,M=C=2)

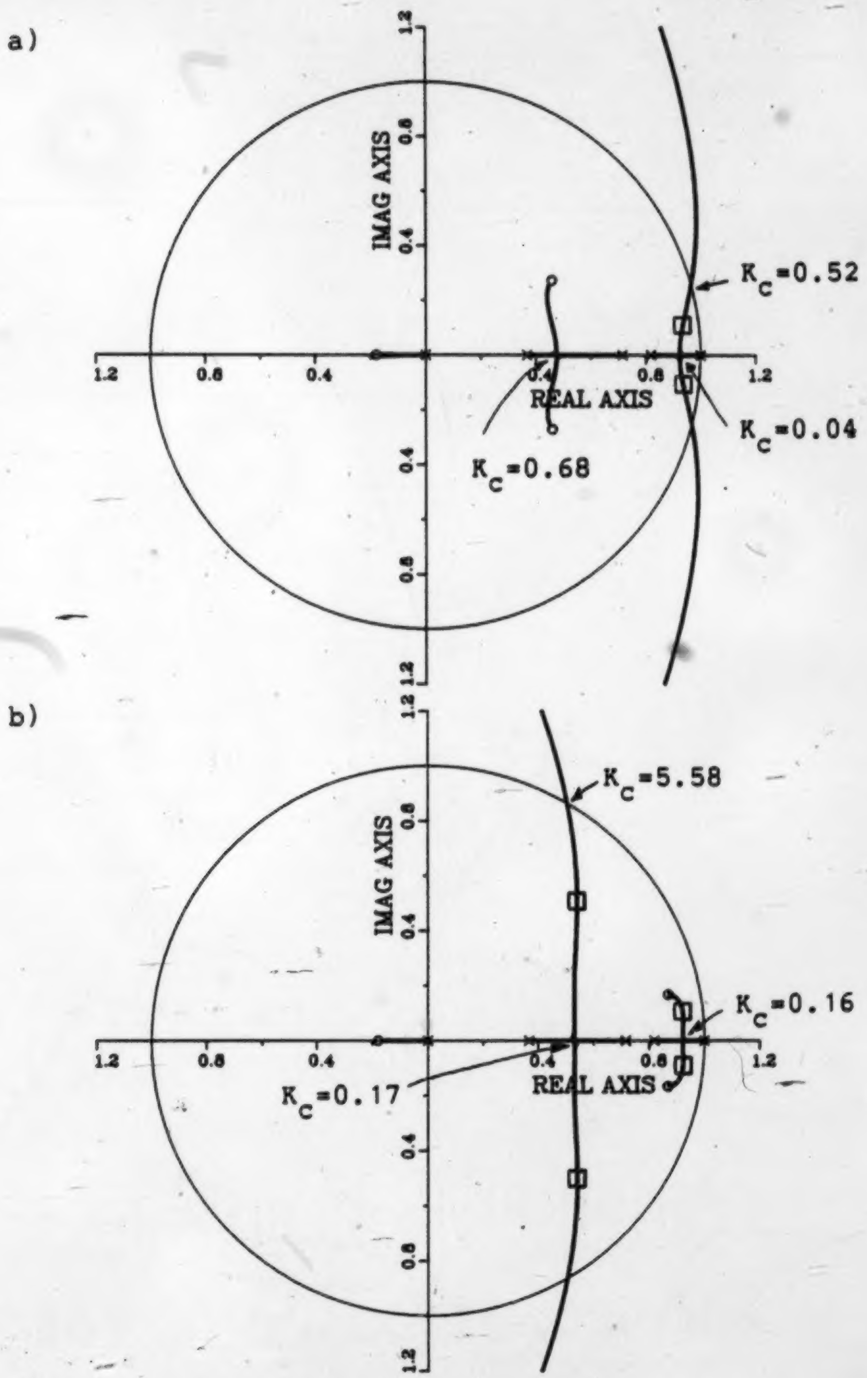

Figure 6.14 Root loci of Process Model 4 with a) GMVPID and b) PPPID controllers (P=3,M=C=2)

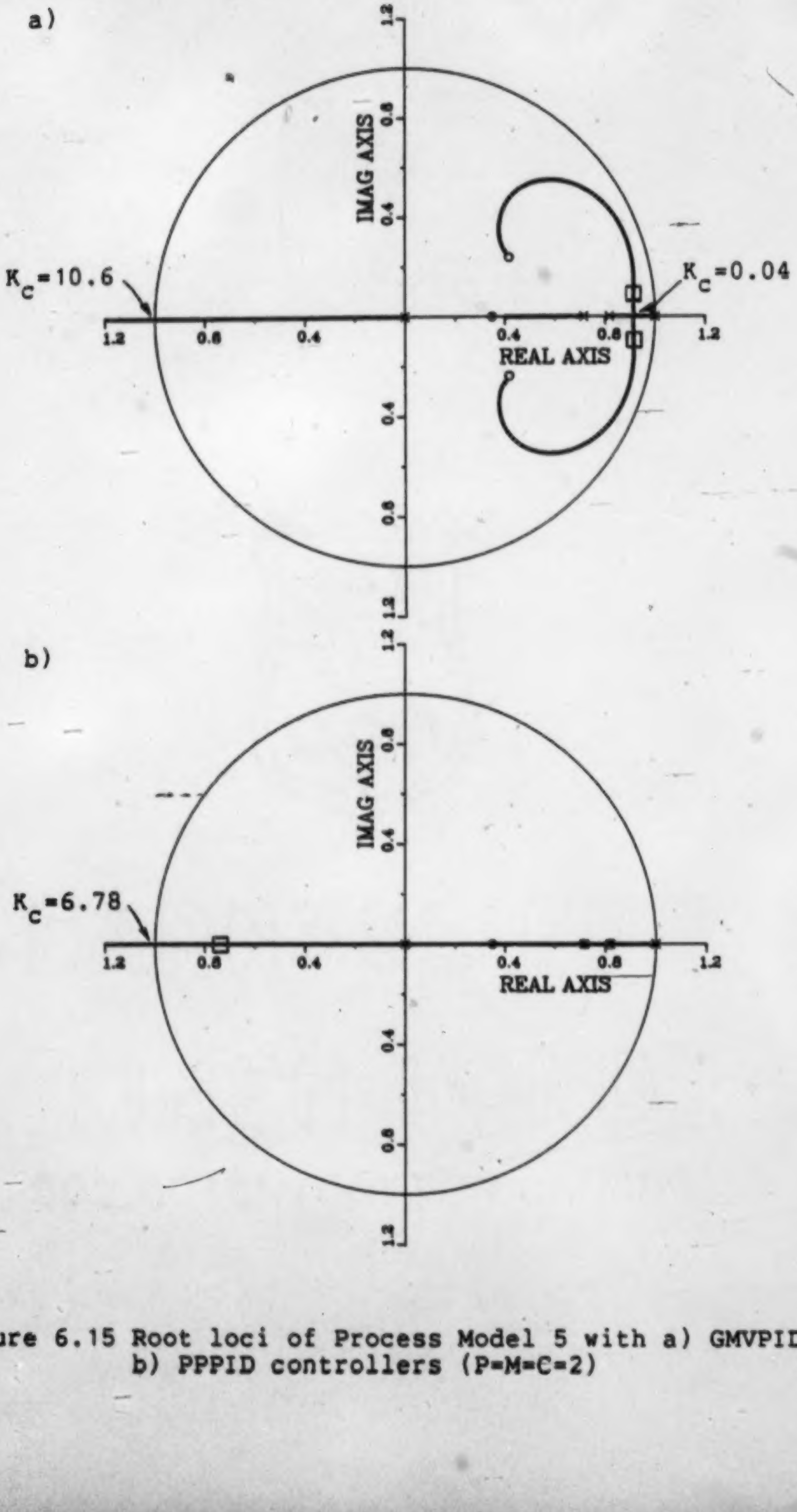

Figure 6.15 Root loci of Process Model 5 with a) GMVPID and b) PPPID controllers (P=M=C=2)

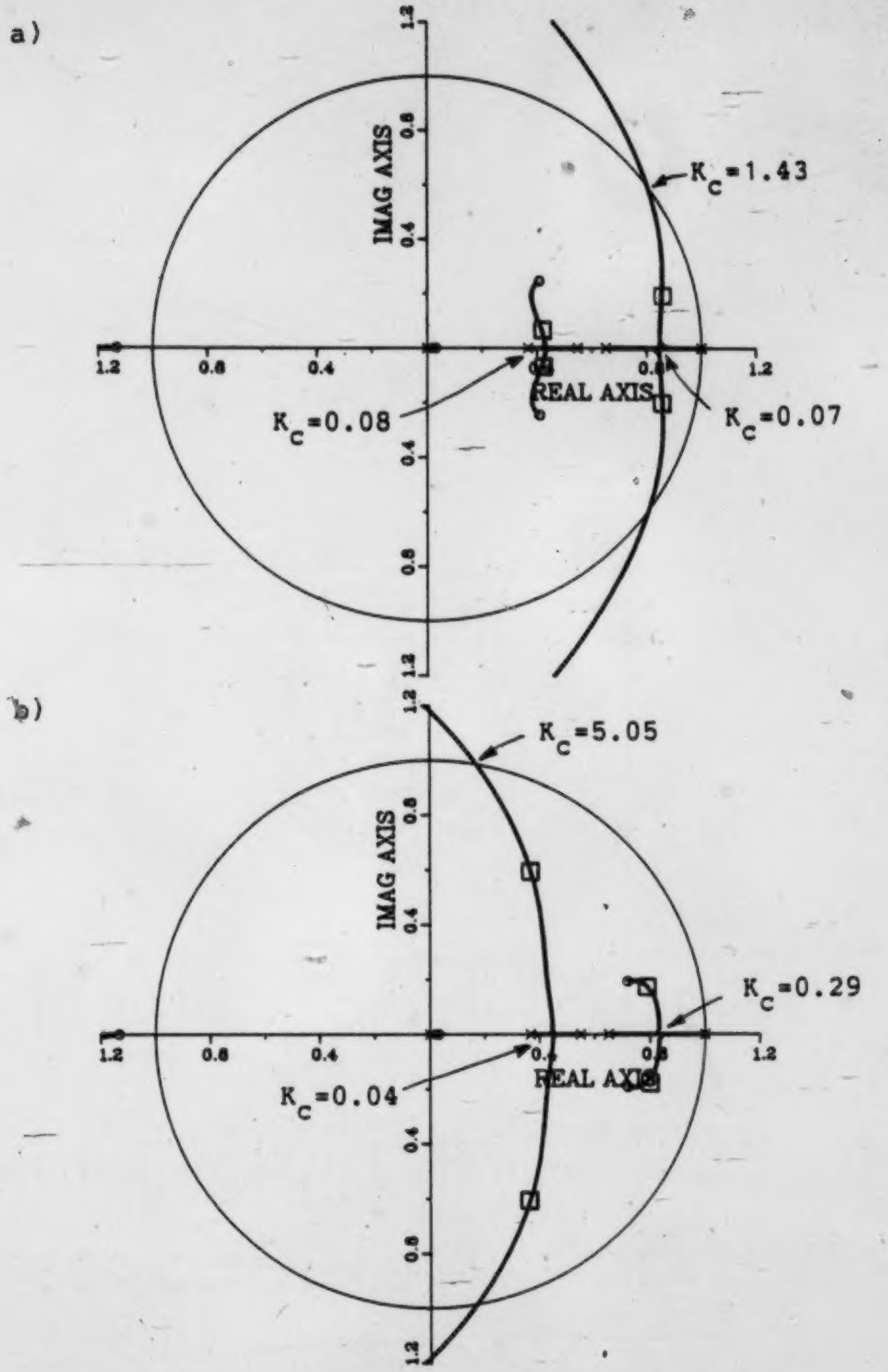

Figure 6.16 Root loci of Process Model 6 with a) GMVPID and b) PPPID controllers (P=3,M=C=2)

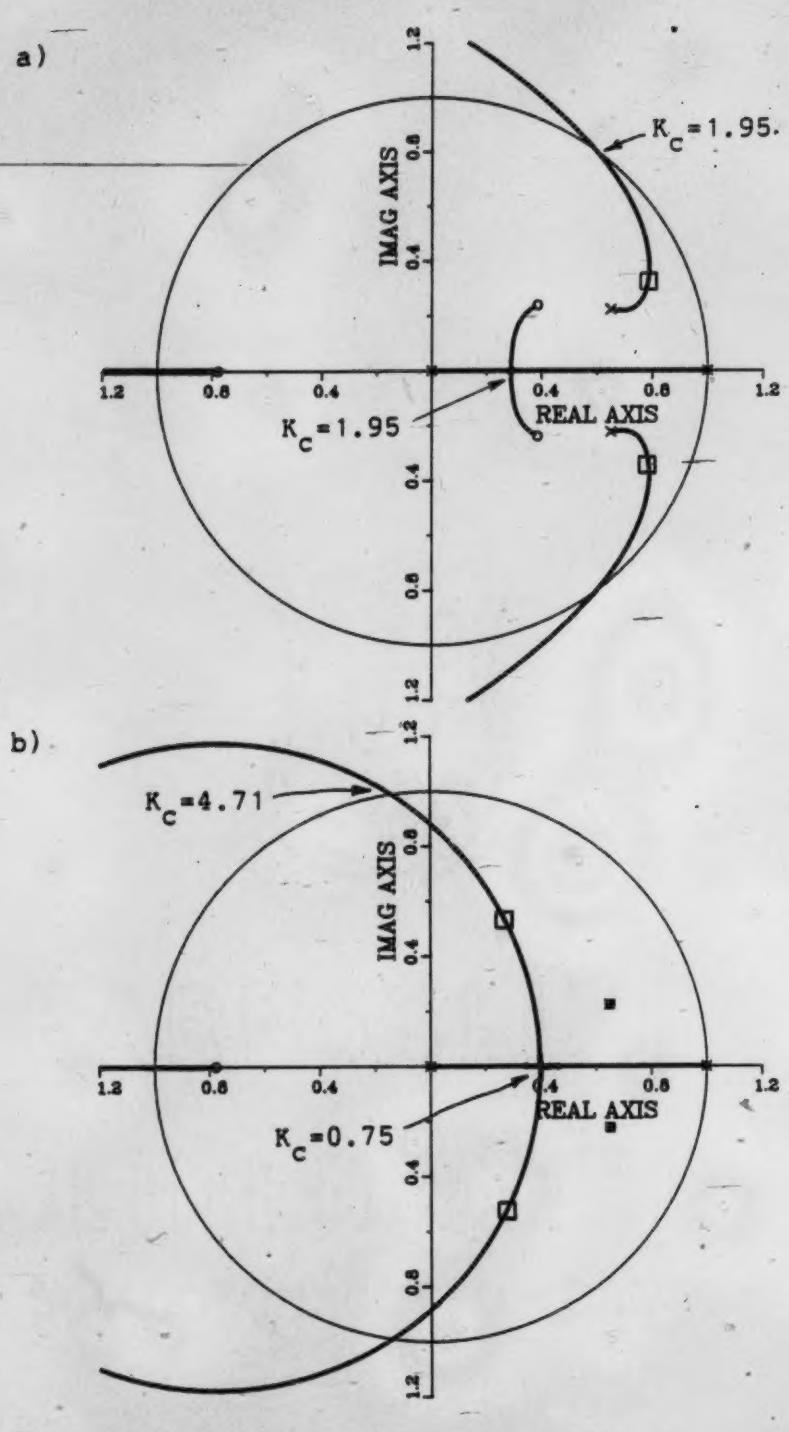

Figure 6.17 Root loci of Process Model 7 with a) GMVPID and b) PPPID controllers (P=M=C=2)

controller gains (K_c<0.07). The main branches are between the controller pole at z=1 and the nearest process pole. These branches, therefore, lie near the unit circle and leave the unit circle at low controller gains for overdamped, second and third order processes with no numerator dynamics. The net effect for all the processes considered, is longer rise and settling times.

The PPPID controller zeroes cancel the estimated process poles. If the process is second order (ideal), this controller cancels the true process poles. This effectively reduces the closed-loop system to the two controller poles and the process zero. The root loci breakaway point is well within the unit circle which results in better rise and settling times (see Figures 6.1b and 6.17b). If the process has numerator dynamics the root locus remain entirely on the real axis. For the mismatch case (P=3, C=2) the controller is unable to cancel the process poles because the estimated model is only second order (M=2). For these cases the PPPID controller calculates complex controller zeroes near z=0.8±j0.2 (see Figures 6.12b, 6.14b, 6.16b) as its estimated second order model is underdamped. This results in very large derivative times (i.e. $T_d > T_i/2$). The better controller performance for the PPPID controller is summarized in Figures 6.2 and 6.13 for an ideal and a mismatch case.

6.6 Summary

The comparison of the GMV and the PP based PID controllers summarized in Tables 6.1 and 6.2 clearly show that the PPPI(D) controller is superior according to the criteria investigated. The major factor for the difference in performance is the location of the controller zeroes. The GMVPID controller places zeroes near z=0.4±j0.2 well away from the z=1 for both the ideal and the mismatch cases. This results in the complex portion of the root loci being very close to the unit circle at z=1. Closed-loop poles in this area correspond to slower rise and settling times. The PPPI(D) controller places controller zeroes near the process poles. This moves the root loci well within the unit circle where improved rise and settling times are possible.

The PPPID controller is designed to cancel the estimated real process poles which generally lie between 0.5<z<0.8 for the chosen sampling rates. For the ideal case the process poles are exactly cancelled by controller zeroes and the closed-loop system response is accurately predicted from the location of the closed-loop poles. This is seen in Tables 6.1 and 6.2. For the PPPI controller with PM1, there is effectively one closed-loop pole and the time-domain response can be exactly determined by the damping ratio, rise and settling time corresonding to that closed-loop pole. For the PPPID controller with PM2, the effective open-loop poles and zeroes which determine the root loci are the controller poles and the process zero at $z=-0.8371$. Again the time-domain response of this system can be determined accurately by the damping ratio (ζ) and natural frequency (ω_n) associated with the closed-loop poles of the system.

For the mismatch cases the PPPI(D) controller locates controller zeroes near $z=1$ (i.e. $z=0.8$ for PI; $z=0.8\pm j0.2$ for PID). The complex controller zeroes for the PID case are a result of the model that is estimated in the presence of process/model mismatch. The actual third order processes are overdamped, but parameter estimation results in an underdamped second order model. The controller zeroes are the complex poles of this underdamped second order model. These zeroes do not cancel the actual process poles and have a significant influence on the time-domain response. They increase the peak overshoot (M_p) and the rise time (t_r) of the closed-loop system. This is seen in Tables 6.1 and 6.2.

The PPPID controller design method has potential problems when used in the presence of process/model mismatch. This design method calculates very large derivative times ($T_d>T_i$). Thus this controller would be very sensitive to the process and measurement noise of a real process, resulting in excessive controller action. Derivative kick would also be severe at setpoint changes. Finally, Franklin and Powell (1980) show that peak overshoot is quite sensitive to complex zeroes near $z=1$. A small variation of controller zero positions can result in a large change in the maximum overshoot of the closed-loop system. Because of these potential problems a modification to the PPPID controller is suggested when controlling an overdamped process in the presence of process/model mismatch. If the derivative time is calculated to be greater than $T_i/4$, it is reset to $T_i/4.1$. This new T_d is then implemented in the control algorithm.

This modification would reduce the potential problems of noise and derivative kick mentioned earlier. It forces the controller zeroes to be real. This is in keeping with the pole placement design method which attempts to cancel the process poles. For an overdamped process the poles are real, therefore the controller zeroes should be real. This modification meets the well-known Ziegler and Nichols tuning rules. Both the transient response method and the ultimative-sensitivity method call for the derivative time to be $T_d=T_i/4$.

The effect of this modification on the criteria used in Section 6.4 and 6.5 is summarized in Table 6.3 for the mismatch cases. It increases the ultimate gain of the closed-loop system but increases the rise and settling time. It in effect degains the response. The most significant change is the slower rise time. The settling time is only marginally greater. However, the small decrease in performance is more that offset by the usefulness of the algorithm on real processes. Figure 6.13

Table 6.3 Comparison of PP and modified PP based PID control laws

			PPPPID					PPPID(modified)				
Process	Mismatch	Figure	K_c^u	T_i	t_r	t_s	Notes	K_c^u	T_i	t_r	t_s	Notes
Model		No.	K_c	T_d				K_c	T_d			
PM3	P=3	6.12	3.24	8.10	8.5	53.9	3,5	3.75	8.10	16.0	58.3	2
	M=2	6.13	1.46	6.92				0.90	1.98			
	C=2	6.18			*	*		*				
PM4	P=3	6.14	5.58	3.84	3.2	36.4	3,5	6.86	3.84	11.8	45.6	2
	M=2	6.19	2.30	4.52				0.67	0.94			
	C=2				*	*		*				
PM6	P=3	6.16	5.05	8.54	7.3	37.7	3,5	5.75	8.54	12.0	45.3	2
	M=2	6.20	1.90	5.04				1.25	2.08			
	C=2				*	*		*				

Notes: 1. no branches in root locus
2. two branches in root locus
3. four branches in root locus
4. circle pattern in root locus
5. complex controller zeroes

compares the time response of this modification with the previous PID controllers. Figures 6.18-6.20 show the effect of this modification on the root loci of these process models. The number of branches has been reduced from four to two and the controller zeroes have been forced to be real.

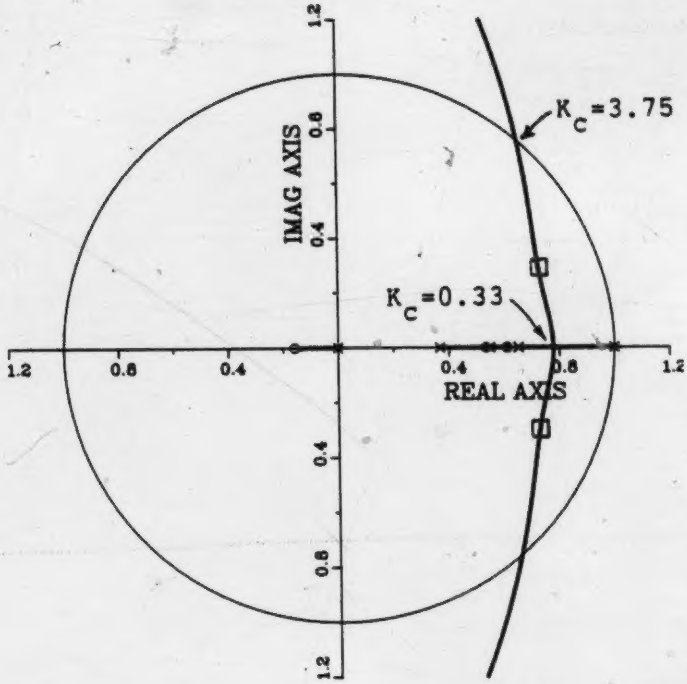

Figure 6.18 Root loci of Process Model 3 with modified PPPID controller (P=3,M=C=2)

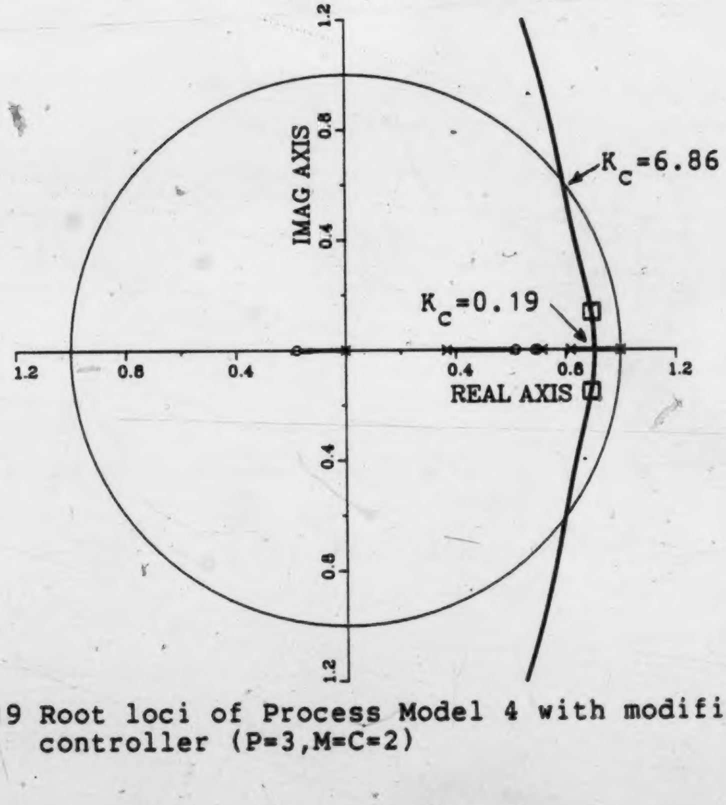

Figure 6.19 Root loci of Process Model 4 with modified PPPID controller (P=3,M=C=2)

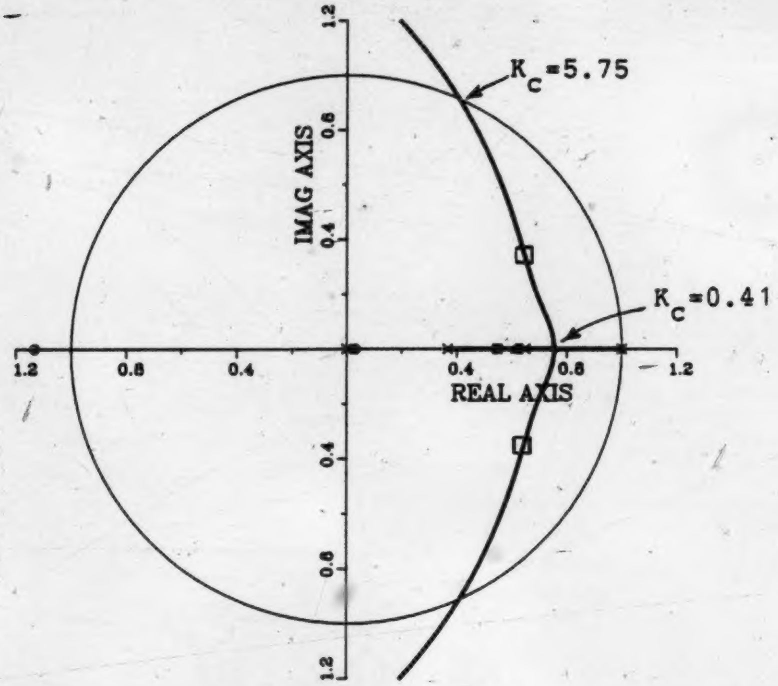

Figure 6.20 Root loci of Process Model 6 with modified PPPID controller (P=3,M=C=2)

7. Performance of Self-Tuning PID Algorithm with Linear Systems: In the Presence of Process/Model Order Mismatch

7.1 Introduction

The analysis of Chapter 6 concluded that the self-tuning PI(D) algorithm derived from a pole placement design criterion gives superior closed-loop performance to setpoint changes. The objective of this chapter and the next is to evaluate the performance of this self-tuning PPPI(D) controller. The evaluation tests the controller over the wide range of linear SISO process models outlined in Chapter 5 for both the PI and PID forms. Performance in the presence of process/model order mismatch, time delays, process bias and deterministic and stochastic disturbances are also considered.

All simulation studies are based on a continuous-time simulation of the process models. The self-tuning PID algorithm used is the modified PPPI(D) algorithm outlined in Chapter 6. The method of evaluation is based on convergence of the estimated process parameters, tuning of the controller constants and servo and regulatory control.

The layout of the chapter is as follows. Section 7.2 discusses the specifics of the self-tuning PI(D) implementation as well as the continuous-time plant simulator used to generate the results. Section 7.3 presents the simulation results for the ideal case of no process/model mismatch. Process/model mismatch is considered in Section 7.4. Summary comments of the PPPI(D) performance are made in Section 7.5.

7.2 Self-Tuning PID Algorithm Implementation

The implementation of an explicit adaptive control algorithm generally involves three steps: process parameter estimation, control law parameter calculation and controller output calculation. A typical control cycle for the self-tuning PPPI(D) controller begins with the explicit estimation of the process parameters for a prespecified process model structure. The estimated process parameters are used to calculate the integral (T_i) and derivative (T_d) times for the PID controller. The estimated process model and the controller parameters are used to set up the characteristic equation of the closed-loop system as outlined in Section 3.6. The damping ratio corresponding to each root of the characteristic equation is calculated. (This calculation assumes a second order process with no numerator dynamics.) The dominant damping ratio is compared with the damping ratio equivalent to the peak overshoot setpoint. The controller gain is incremented or decremented to meet this maximum overshoot specification. The current control signal is calculated based on the calculated K_c, T_i and T_d after they have been filtered by a first order digital filter.

The control software for the simulation studies is designed to allow numerous identification and control options. For parameter estimation the user has the choice of first or second order model identification, incremental or positional variable identification, and recursive least squares or recursive extended least squares estimation algorithm. The controller can be chosen as first (PI) or second (PID) order. If a second order process model is estimated, model reduction to a first order model is done before the PI controller is designed. The PPPI(D) implementation can handle a reasonable amount of time delay. The time delay can be known or unknown. If unknown, the minimum and maximum estimated time delay is used to increase the order of the $B(z^{-1})$ polynomial. A unique feature of the PPPI(D) control algorithm is that it can be tuned online. If the response to a setpoint change is sluggish, the M_p specification can be increased online. This causes the controller gain to be increased. For the study of processes dominated by time delays, the adaptive time delay compensator (TDC) of Vogel and Edgar (1980) is implemented in the simulation software with many of the options listed above.

Self-tuning controllers require initialization of numerous parameters (e.g. initial process parameter estimates, variable forgetting factor parameters and initial covariance matrix, etc.). The individual effect of many of these parameters is not investigated in this study, and therefore are fixed at the same value for the majority of runs. Table 7.1 summarizes those parameters which are fixed

Table 7.1 Initialization parameters for self-tuning PI(D) controller

Initialization Parameters	Value or Option
Intial process parameter estimates	$a_1 = -0.9$ $a_2 = 0.0$ $b_0 = 1.0$ $b_1 = 0.0$ $c_1 = 0.0$
Variable forgetting factor parameters	$\alpha = 0.8$ $K = 1.0$ $\lambda_f = 1.0$ $\lambda_{min} = 0.99$ $\lambda_{init} = 0.99$
Intial covariance matrix	$P_0 = 1000I$
Control law parameters and limits	$K_c$min = 0.025 $K_c$max = 20.0 Filter = 0.7 no high, low or rate limits
Estimation variable type	Incremental
Maximum overshoot setpoint	$M_p = 15\%$

and the values they are assigned. If one of these initialization parameters is changed for a particular run, it is noted.

The simulation study results are based on a continuous-time simulation of the linear process. The transfer function models of the processes are transformed into differential equations and integrated. The integration method is the Euler predictor-corrector method. The integration step is chosen as $T_s/100$. Deterministic disturbances are implemented in the same manner. Stochastic disturbances are handled by digital techniques. A discrete white noise sequence is filtered by the noise model $C(z^{-1})/A(z^{-1})\Delta$. The filtered output N(t) is then added to the output at every sample interval.

All simulations were done on the University of Alberta Amdahl computer. Source listings of the simulation software are available from the DACS Center or by contacting Dr. S.L. Shah.

7.3 Simulation Results for No Process/Model Mismatch

Mismatch here is defined as a difference between the order of the process and the order of the controller. For no mismatch to be present the order of the process must be equal to the order of the model used to design the self-tuning controller. For example if the process is second order, the order of the controller must be second order (i.e. based on a second order model). Four ideal cases are possible from the linear processes listed in Chapter 5. The results of these cases are now considered.

The simplest no mismatch case is the first order Process Model 1 (PM1) with a self-tuning PPPI controller. The results of this simulation are shown in Figure 7.1. The initial parameters and options are as outlined in Table 7.1. Note the excellent servo control to setpoint changes. The controller tunes in quickly with the estimated first order process parameters converging to within the third decimal point of the true values after 10 sampling intervals. The control effort required at setpoint changes is large in keeping with the "setpoint-on-P&I&D" controller structure. The time-domain response of M_p=15% is exactly as specified in the z-domain.

Figure 7.2 shows the base case results for an overdamped second order process with no numerator dynamics (PM2) and a PID controller. Servo control is excellent even while the controller is tuning. A maximum overshoot specification of M_p=15% is realized in the time-domain. Convergence of the estimated process parameters to the true parameter values is smooth and uniform. The estimated parameters move only at setpoint changes when there is dynamic information. The estimated parameters converge to within 0.01 of the true values after 100 sampling intervals. The controller constants (K_c,T_i,T_d) converge slowly as well. Integral and derivative action increase as the estimated process parameters converge to the

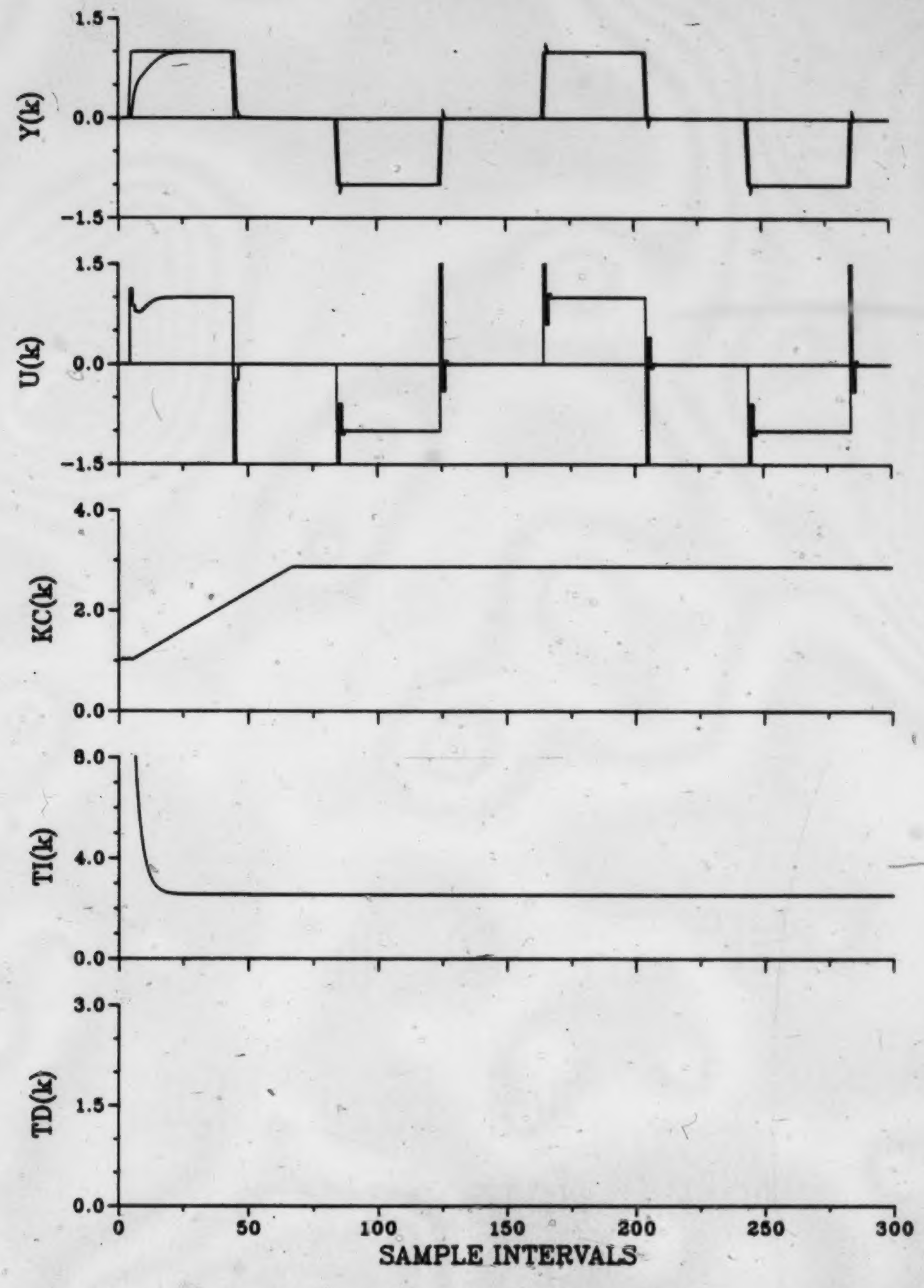

Figure 7.1 Servo self-tuning PI control of Process Model 1 (INC,P=M=C=1,$M_p$sp=15)

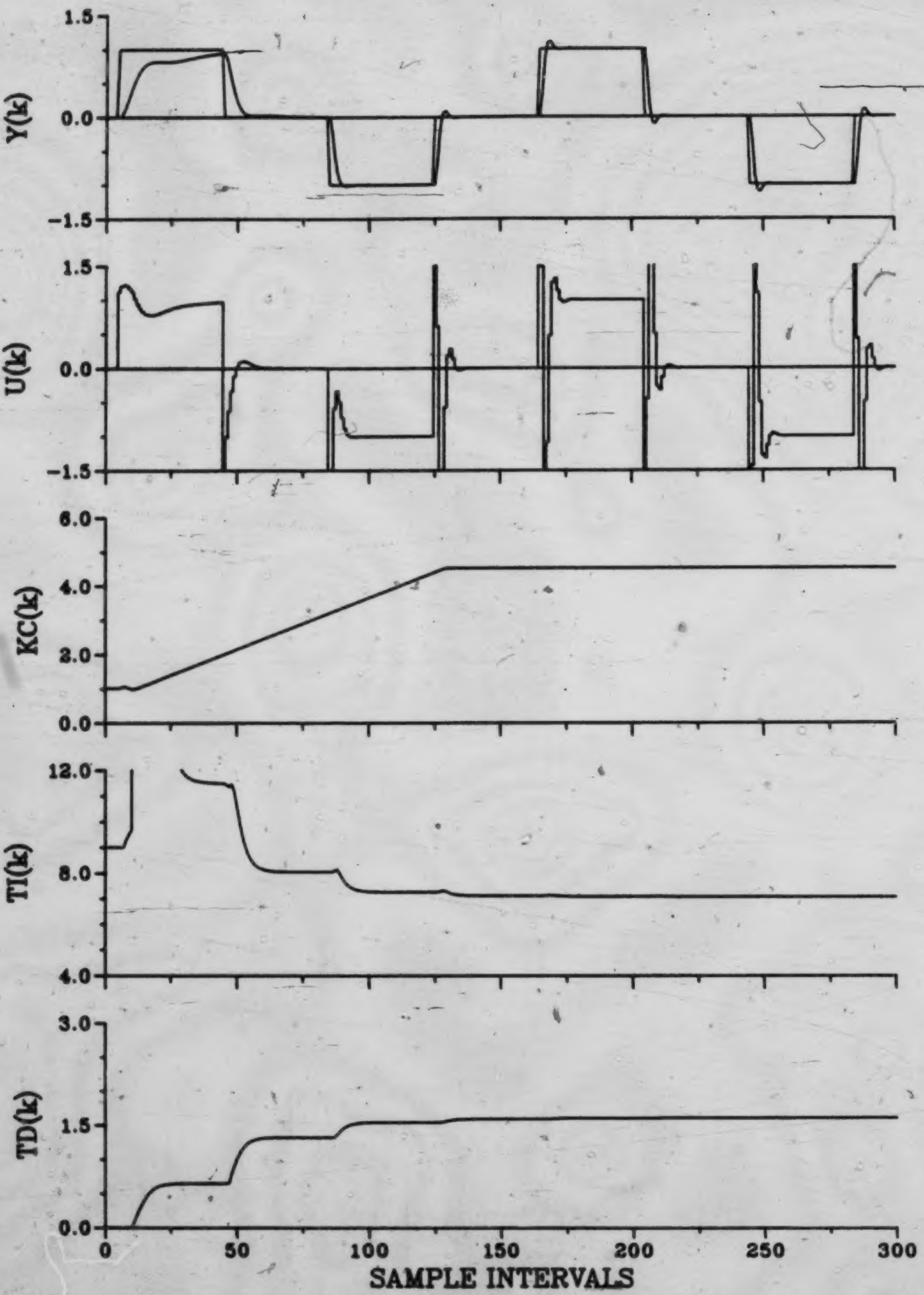

Figure 7.2a Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15)

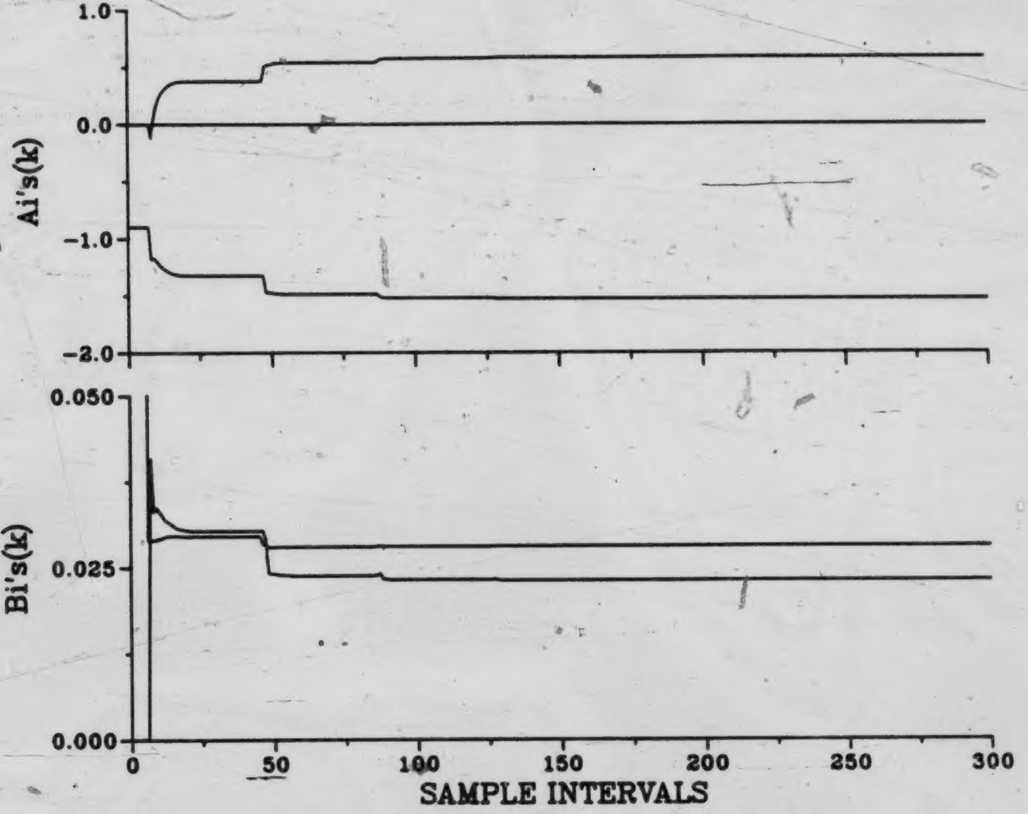

Figure 7.2b Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15)

true values. The controller gain increases as the parameters converge. This is a result of the fixed increment K_c can move per sampling interval. In addition all controller constants are filtered by a first order digital filter with a filter constant equal to 0.7. The control effort at setpoint changes is large.

The results for an overdamped second order process with numerator dynamics (PM5) are shown in Figure 7.3. These results show the ideal base case for processes with a minimum phase zero in the continuous-time domain. The servo response is good but not as fast as the previous case. In addition the output response does not show an overshoot as specified in the z-domain. This is a result of the location of the discrete-time zero in the z-plane at (0.35,j0.0). The root locus plot of this system reveals that the root loci always lie on the real axis. The only way this closed-loop system can have an actual overshoot of M_p=15% is to have a ringing pole at (-0.71,j0.0), which is clearly undesirable. The convergence of the estimated process parameters and controller constants is generally as before. The tuning of K_c exhibits a peak as does T_i. This is due to the way the estimated process parameters converged during the initial transients.

The last no mismatch base case is shown in Figure 7.4 for an underdamped second order process with no numerator dynamics (PM7). Servo control performance is excellent. The time-domain response shows the M_p=15% as

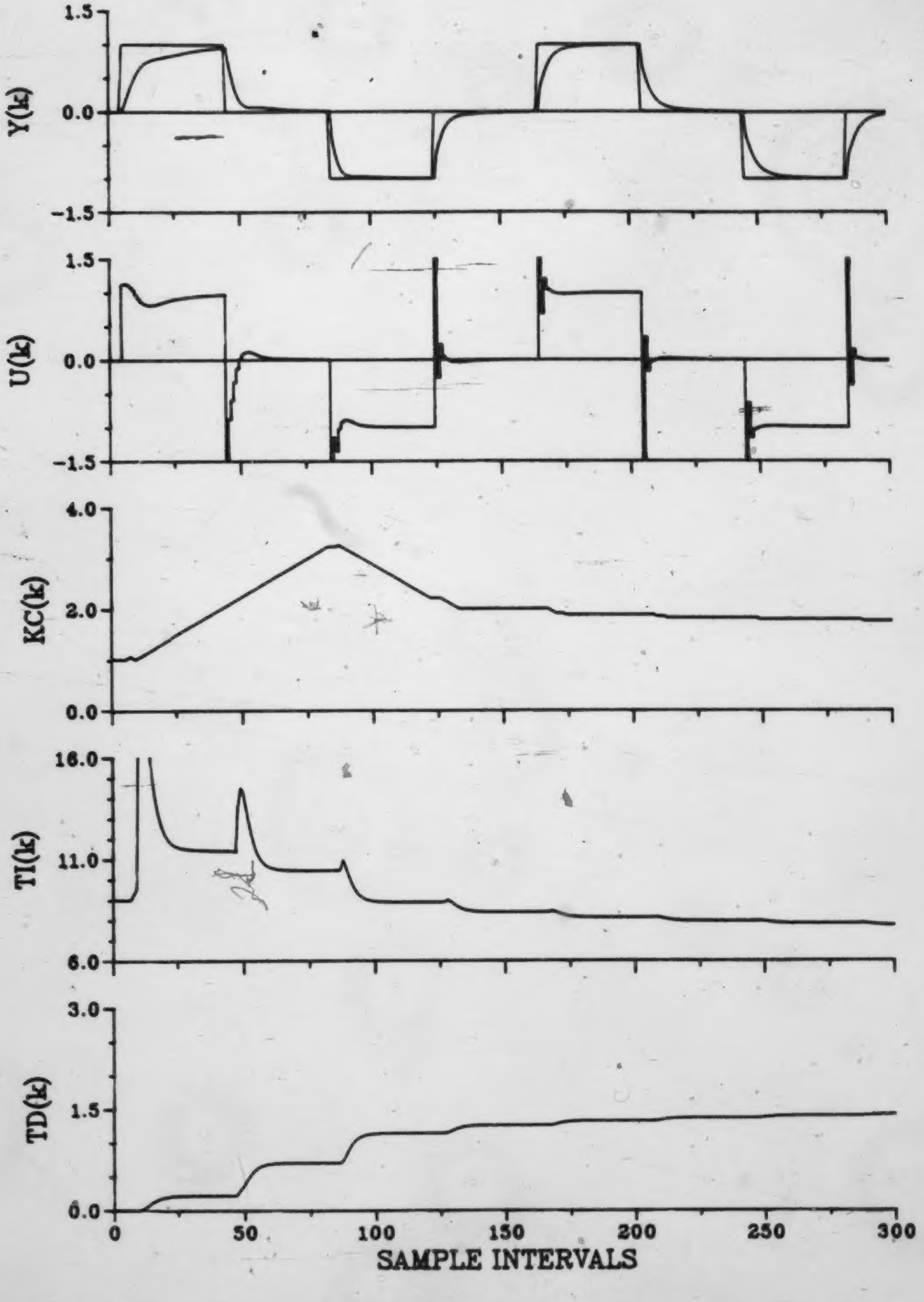

Figure 7.3 Servo self-tuning PID control of Process Model 5 (INC,P=M=C=2,$M_p$sp=15)

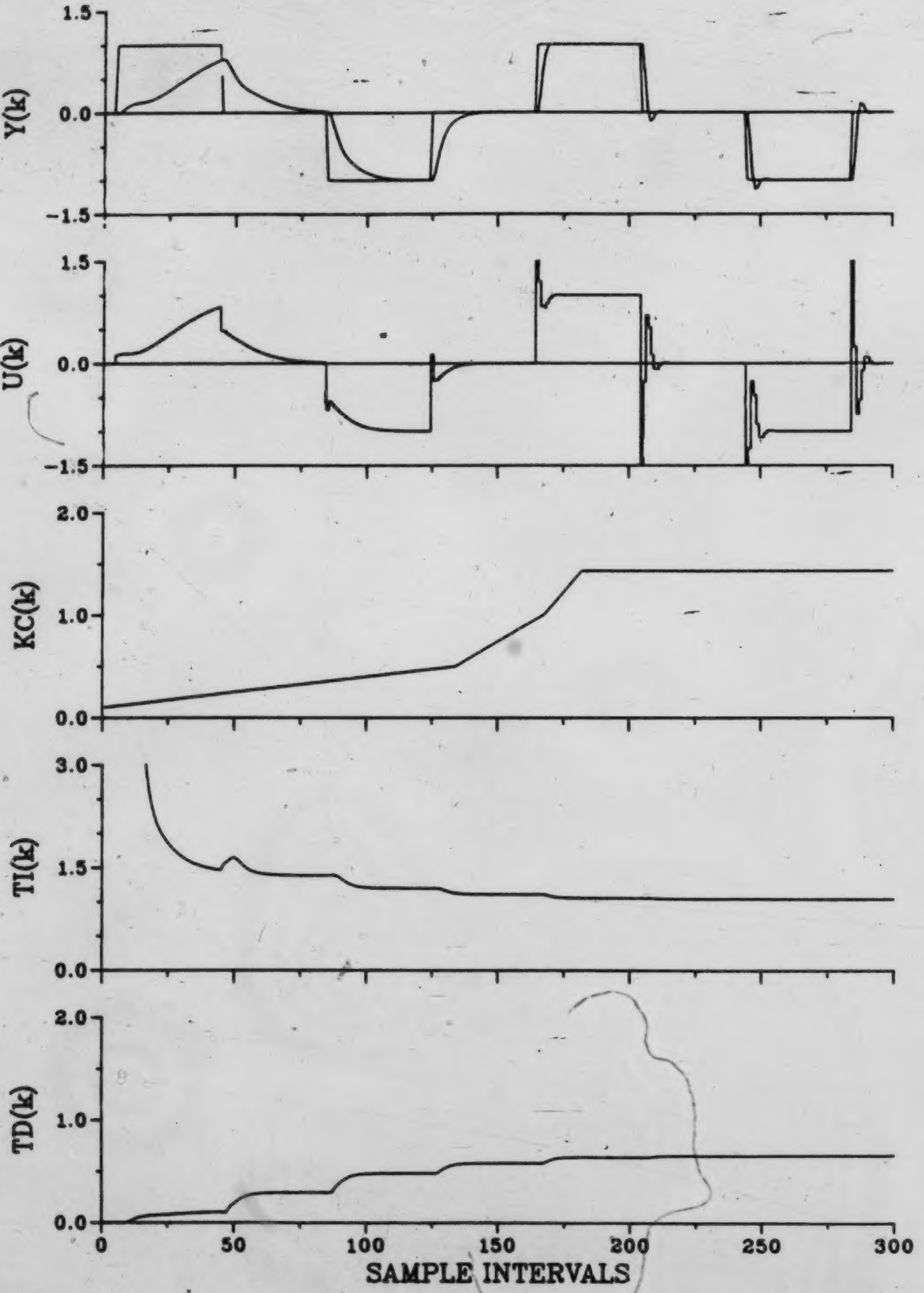

Figure 7.4 Servo self-tuning PID control of Process Model 7 (INC,P=M=C=2,$M_p$sp=15)

specified by the damping ratio in the unit circle of z-plane. Convergance of the estimated process parameters is smooth and within 0.03 of the true values after 150 sample intervals. The integral ~~and~~ derivative times move similarly, increasing their respective control action as they converge. The tuning in of the controller K_c is slow in this case. This is a result of a very conservative initial estimate of K_c $(K_c=0.1)$. If $K_c(\text{init})=1.0$ had been chosen, convergence of all parameters would have been more rapid.

A unique feature of the self-tuning PPPI(D) control algorithm is that it allows for online shaping of the closed-loop response. Since the maximum overshoot setpoint determines the controller gain, it can be changed online. This then modifies K_c to produce the desired response. This is illustrated in Figure 7.5 for PM2. The M_p setpoint is varied from 15-30% in 5% increments. Since this is a no mismatch case with no numerator dynamics, the time-domain response is as specified. Tuned process parameter estimates are used to initialize this run.

In summary these no mismatch runs show the characteristics of the self-tuning PPPI(D) controller for the ideal case. Estimated process parameter convergence is uniform and smooth to the true values. Parameter estimates are updated only during dynamic transients of the process. This is due to incremental variables identification. The tuning of the controller constants is

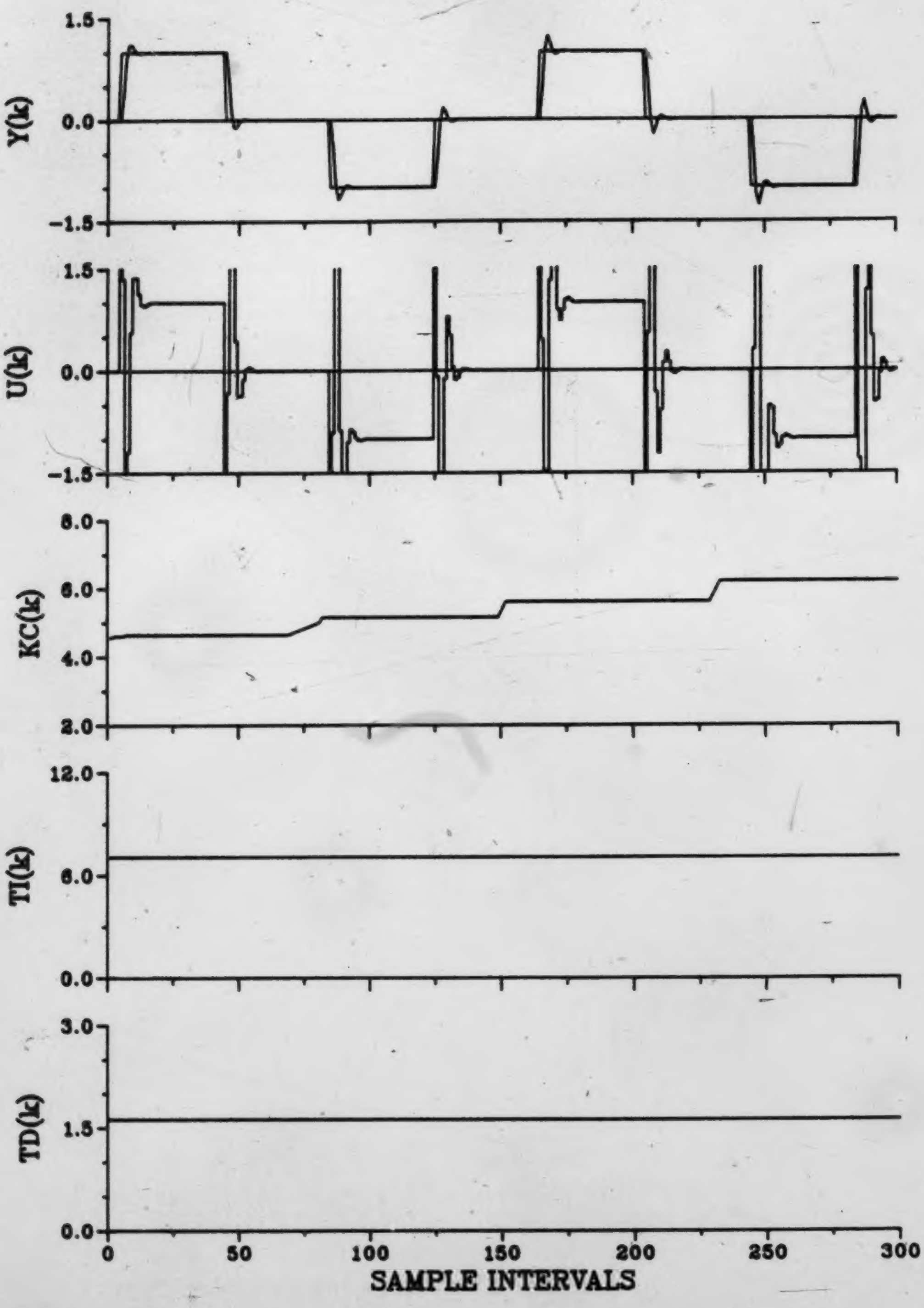

Figure 7.5 Online Mp setpoint (15-30%) tuning of self-tuning PID controller for Process Model 2

also smooth with amount of control action increasing as the estimated process parameters converge. The closed-loop system exhibits the maximum overshoot as specified in the z-domain for processes with no numerator dynamics.

7.4 Simulation Results for Process/Model Mismatch

The performance of the self-tuning PPPI(D) controller is now examined for mismatch between the process order, model order and controller order. For example, if a PI controller (i.e. a first order controller) is used to control a third order process, a mismatch of 2 exists. This section is divided into two parts. The first part considers the self-tuning PPPI controller with those linear process models which result in a mismatch. The self-tuning PPPID controller is then evaluated in the presence of process/model mismatch in the second part. Both parts also look at online tuning of the M_p setpoint for the mismatch case. It is noted here that all the results presented in this section are based on parameter estimation of a second order model. If a PI controller is designed, the estimated second order model is reduced to a first order model by Warwick's model reduction technique outlined in Section 4.4.

7.4.1 PI Control and Mismatch

Self-tuning PI control results in process/model mismatch for seven of the linear processes outlined in Chapter 5. The simulation results of these combinations are summarized in Table 7.2. The table lists for each run, the process, the degree of mismatch, the M_p setpoint and the time-domain response of the closed-loop system for the tuned PI controller. Figures 7.6 - 7.12 show the simulation results for the seven runs. Each figure shows the tuning period and the resultant controller performance. Initialization parameters are as per Table 7.1.

Figures 7.6 - 7.12 show many of the characteristics of the no mismatch cases examined previously. Tuning of estimated process parameters is generally smooth and uniform with all movement of parameters occuring at setpoint changes. The convergence of the controller parameters, K_c and T_i, is similar to the mismatch case. Figures 7.7, 7.8 and 7.10 show the results of process parameter estimation of a second order model when the actual process is third order. The estimated parameters converge to a set of nonunique values (cf. Figure 7.8b).

The results presented in Table 7.2 are based on a maximum overshoot specification of M_p=15%. Only the time-domain response of PM5 and PM8 come close to this specification. Four of the processes exhibit a heavily overdamped closed-loop response. In the presence of mismatch the M_p setpoint becomes a tuning knob to shape the closed-loop response and does not actually specify exactly what the time-domain response will be.

The use of the M_p setpoint as a tuning knob is

Table 7.2 Performance of self-tuning PI controller in the presence of process/model order mismatch

Process Model	Mismatch	Figure No.	M_p SP	Time-Domain Response M_p	t_r	t_s	Comments
PM2	P=2 M=2 C=1	7.6	15%	6%	7.5	27.7	Control action moderate, underdamped response with long settling times
PM3	P=3 M=2 C=1	7.7	15%	0%	75.0	117.0	Control action sluggish, heavily overdamped response. Increase M_pSP
PM4	P=3 M=2 C=1	7.8	15%	0%	30.3	46.0	Control action sluggish, heavily overdamped response. Increase M_pSP
PM5	P=2 M=2 C=1	7.9	15%	16%	2.5	13.8	Control action vigorous, fast underdamped response
PM6	P=3 M=2 C=1	7.10	15%	0%	48.6	81.0	Control action sluggish, heavily overdamped response. Increase M_pSP
PM7	P=2 M=2 C=1	7.11	15%	0%	8.6	13.5	Control action sluggish, heavily overdamped response. Increase M_pSP
PM8	P=3 M=2 C=1	7.12	15%	9%	0.7	2.7	Control action vigorous, fast underdamped response

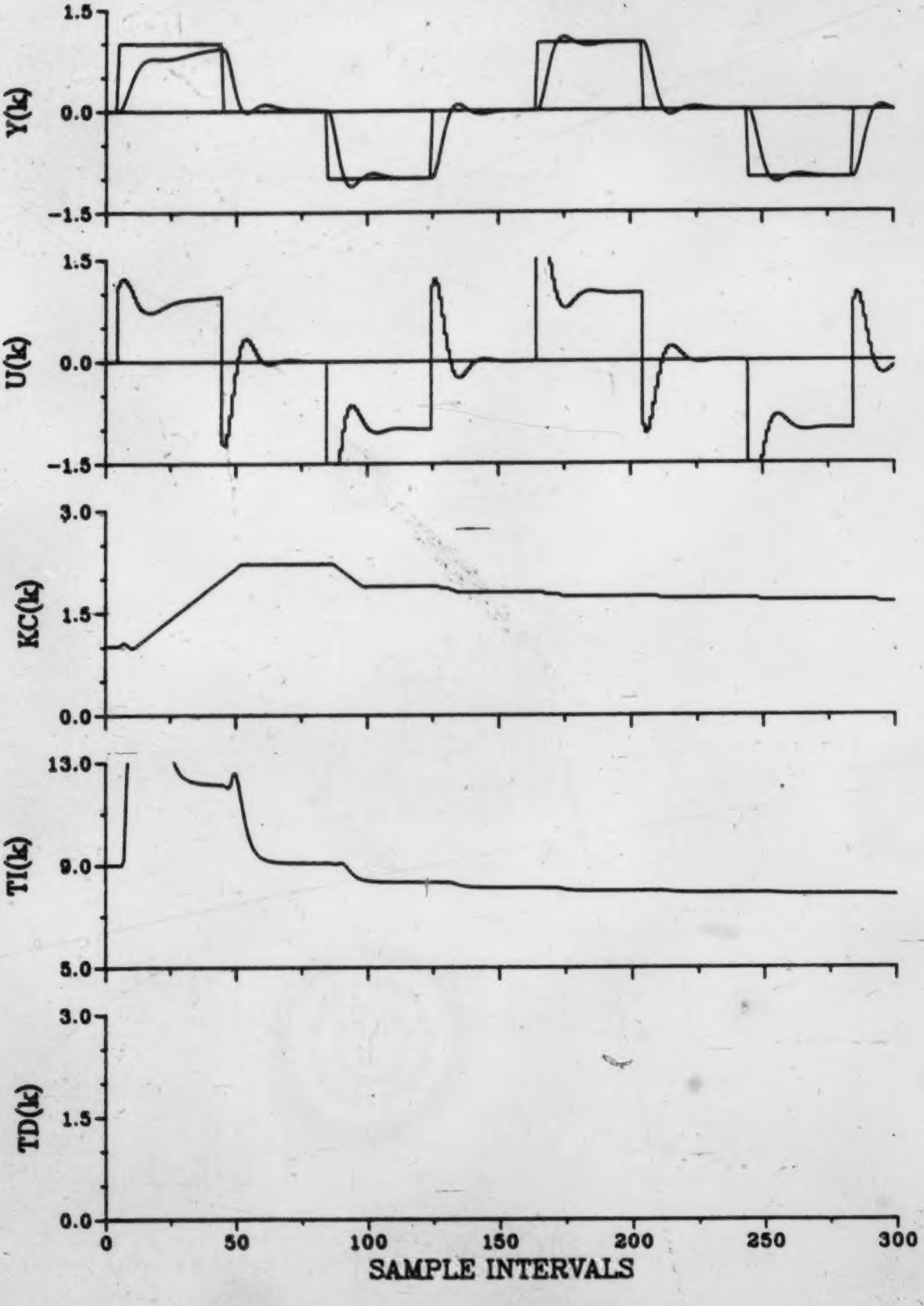

Figure 7.6 Servo self-tuning PI control of Process Model 2 (INC, P=M=2, C=1, $M_p$sp=15)

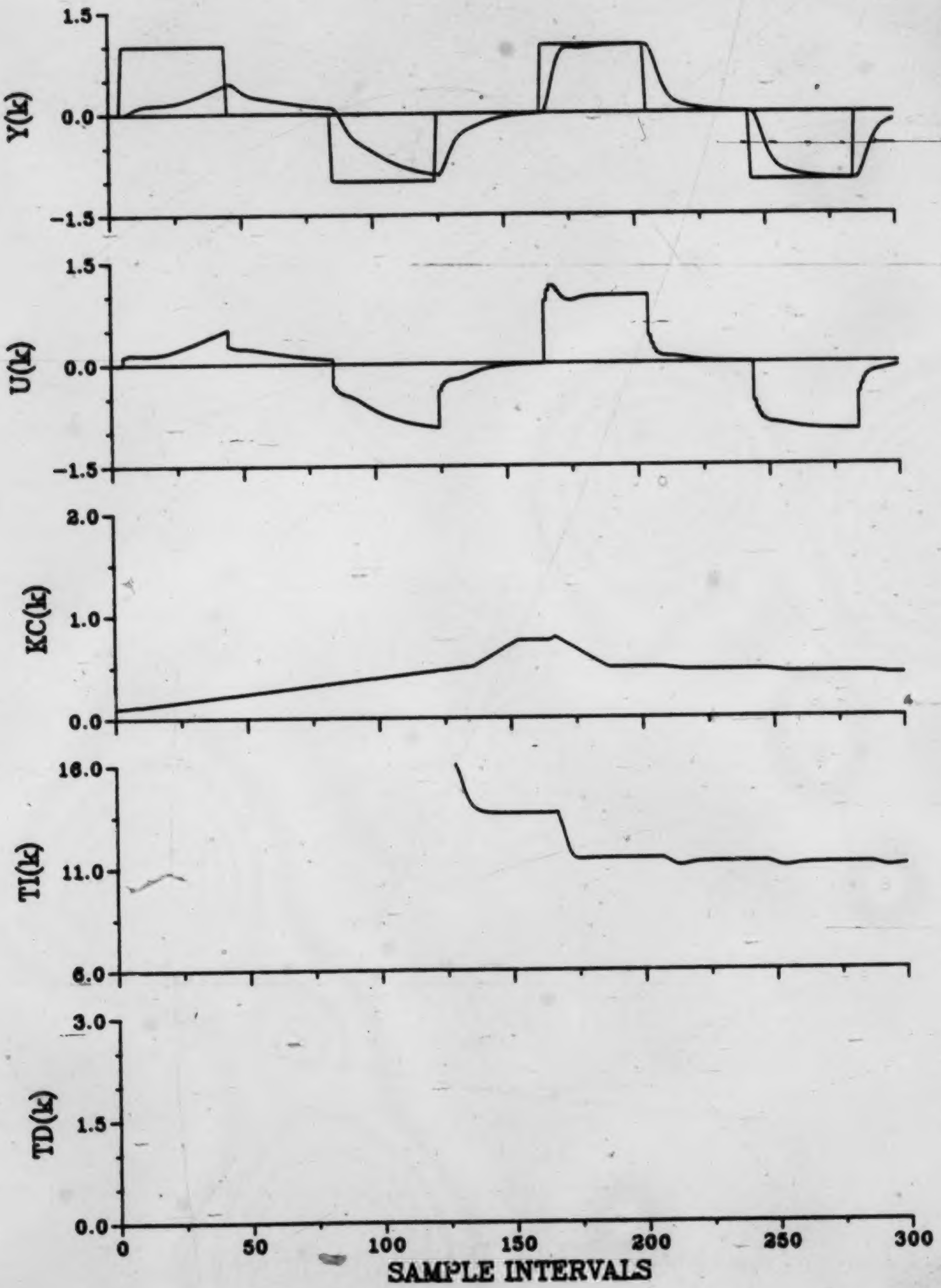

Figure 7.7 Servo self-tuning PI control of Process Model 3 (INC,P=3,M=2,C=1,$M_p$sp=15)

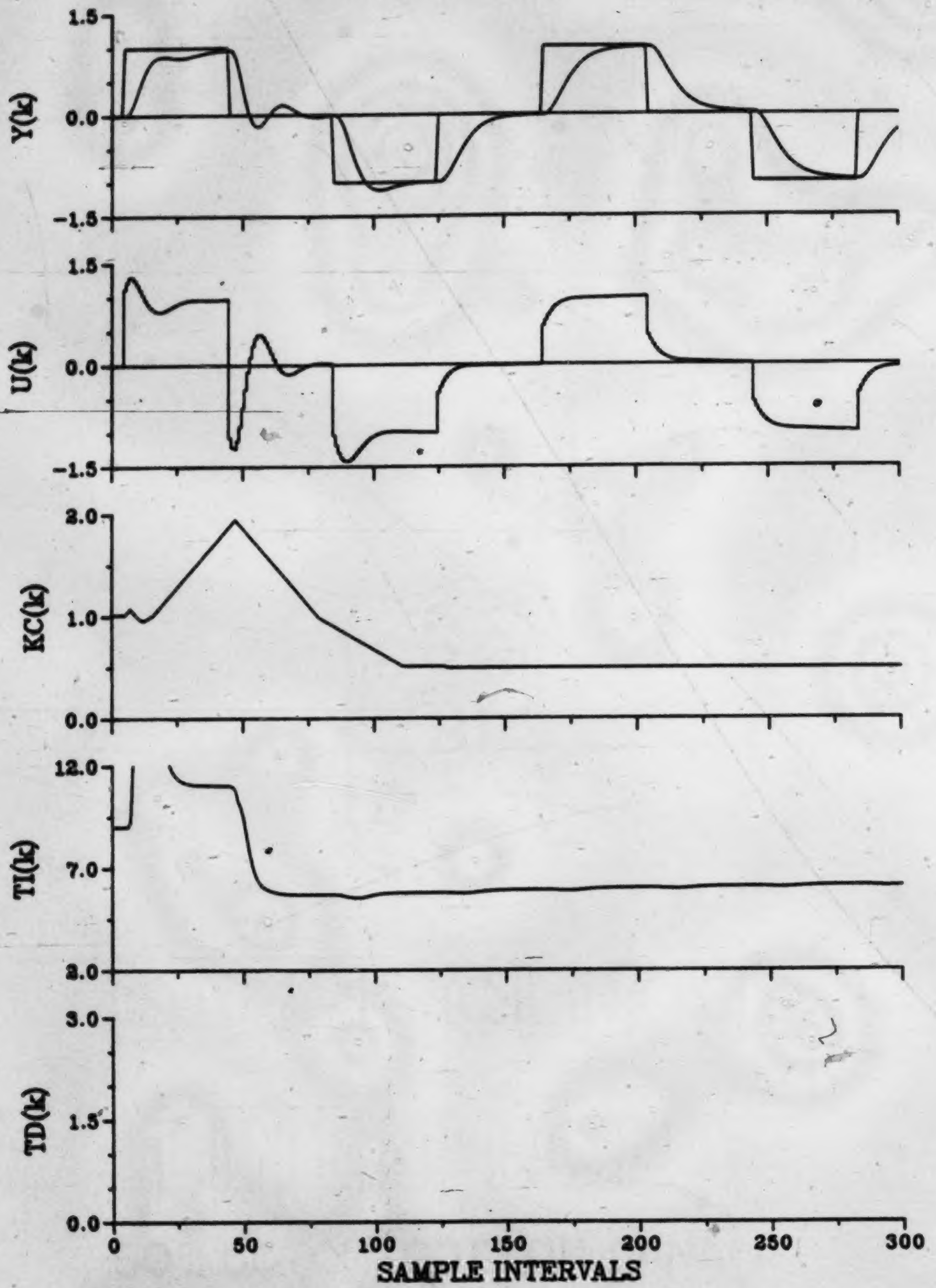

Figure 7.8a Servo self-tuning PI control of Process Model 4 (INC,P=3,M=2,C=1,$M_p$sp=15)

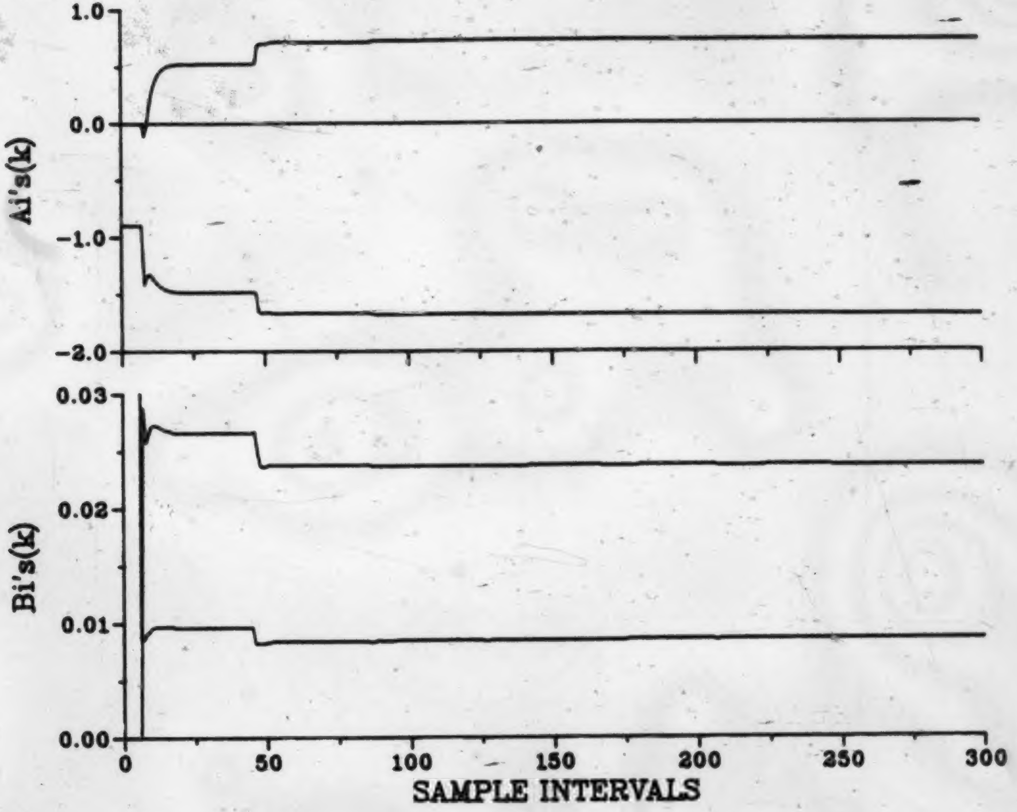

Figure 7.8b Servo self-tuning PI control of Process Model 4 (INC,P=3,M=2,C=1,$M_p$sp=15)

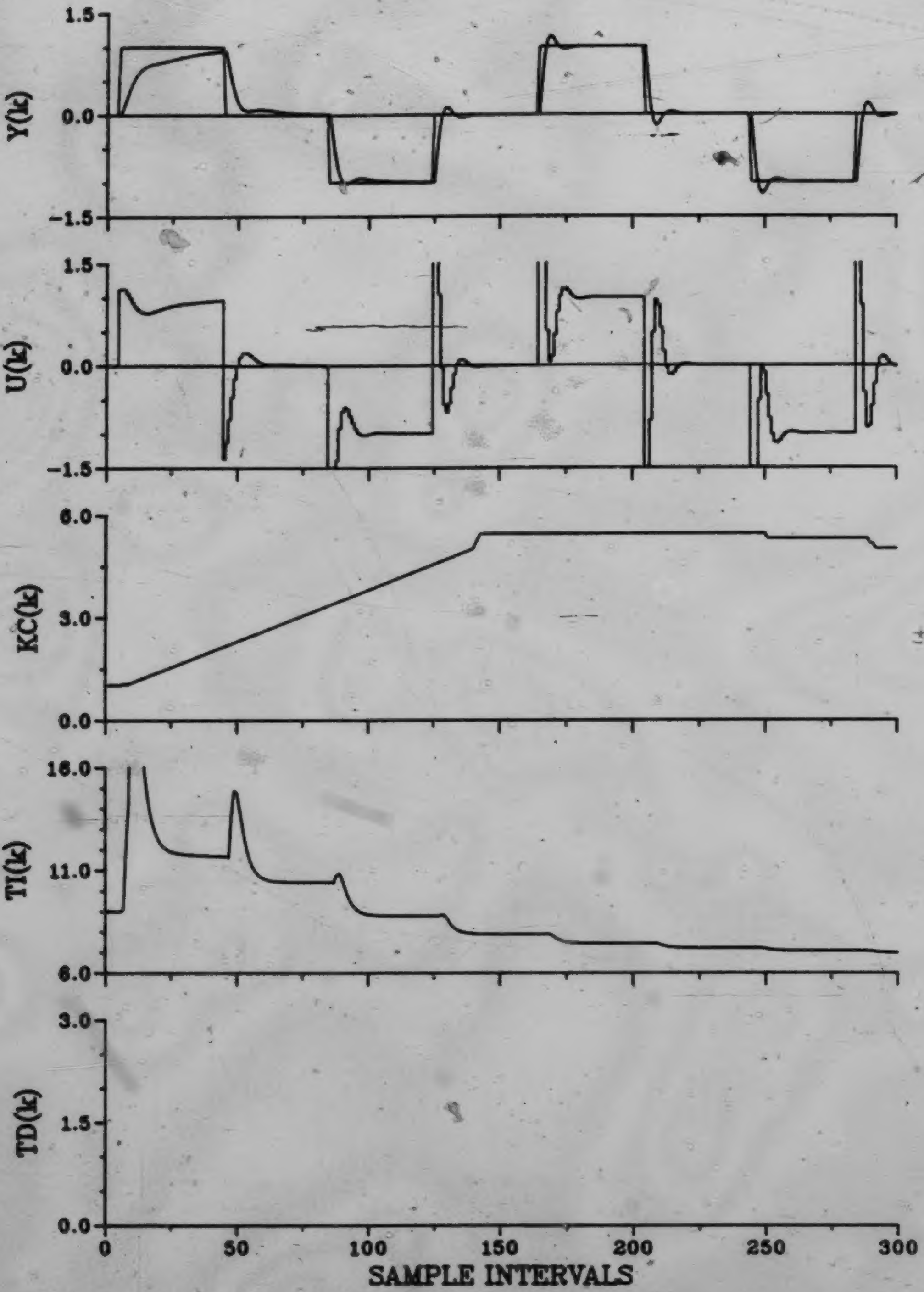

Figure 7.9 Servo self-tuning PI control of Process Model 5 (INC,P=M=2,C=1,$M_p$sp=15)

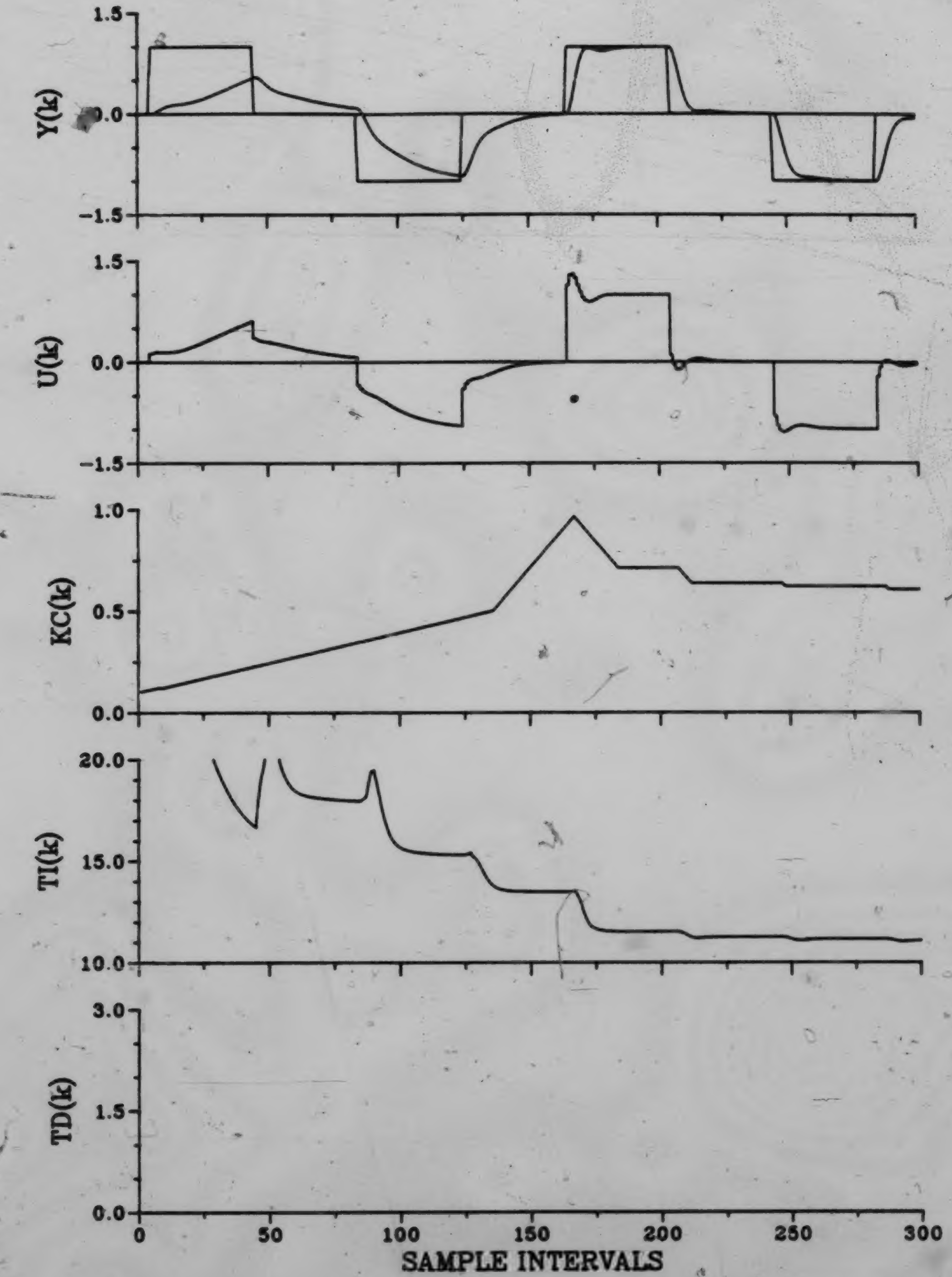

Figure 7.10 Servo self-tuning PI control of Process Model 6 (INC,P=3,M=2,C=1,$M_p$sp=15)

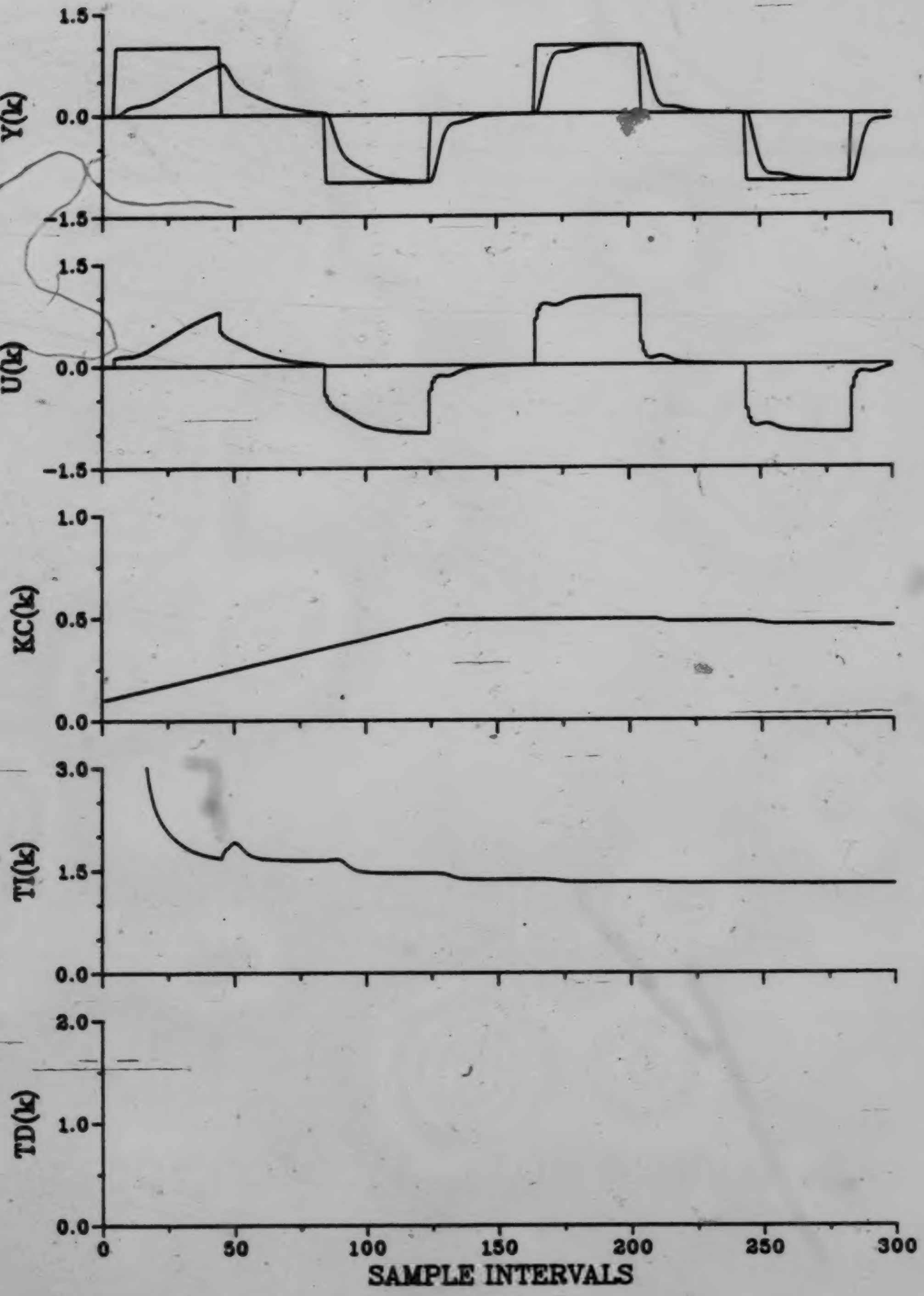

Figure 7.11 Servo self-tuning PI control of Process Model 7 (INC, P=M=2, C=1, $M_p$sp=15)

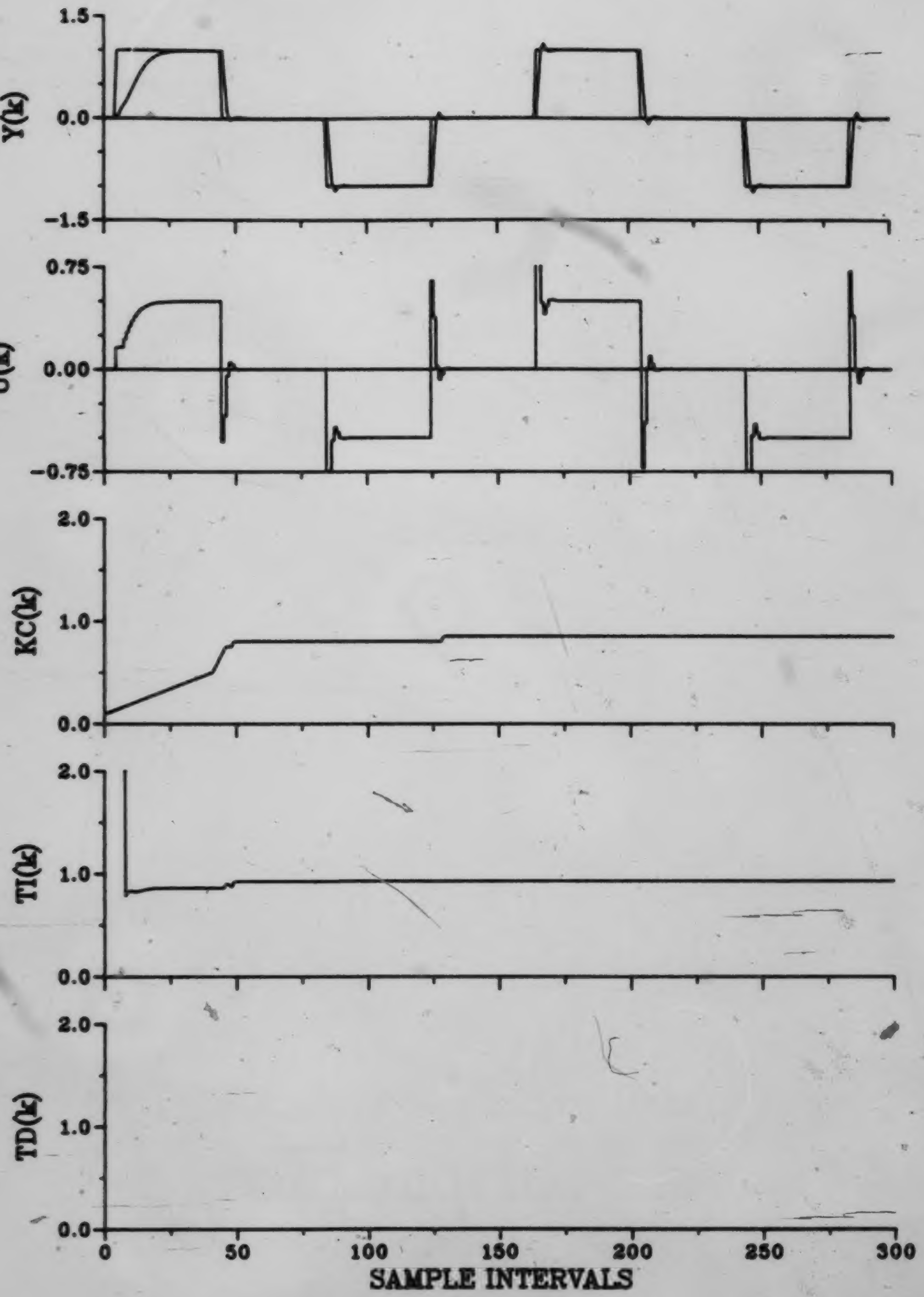

Figure 7.12 Servo self-tuning PI control of Process Model 8 (INC,P=M=2,C=1,$M_p$sp=15)

illustrated in Table 7.3 for two third order processes. The use of these processes results in a mismatch of two between controller order and process order. The M_p setpoint is varied from 15% to 30% in 5% increments. The effect of these increases is to improve the system closed-loop response. For PM3 both the rise time (t_r) and settling time (t_r) decrease with increasing M_p. However, the response remains overdamped. The effect of increasing the M_p setpoint for PM4 is similar. For $M_p^{sp}=30\%$ the response becomes underdamped with an actual $M_p=4\%$. This improves the rise time but causes the settling time to increase over the $M_p^{sp}=25\%$ case. Figures 7.13 and 7.14 show the transient response for these runs. Both runs are initialized with tuned parameter estimates.

In summary the self-tuning PPPI controller performs well in the presence of process/model order mismatch. The M_p setpoint becomes a tuning knob which can be used to shape the closed-loop response. It does not accurately determine the actual closed-loop M_p. For the runs considered here, mismatch causes a detuned controller gain to be calculated. A more desirable response is achieved by increasing the M_p^{sp}.

The runs presented in this section are all based on an estimated second order process model. This gives enough information about the process to do a proper controller design. Controller design based on an estimated first order model when the actual process order is 2 or greater was

Table 7.3 Online tuning of M_p setpoint for self-tuning PI control

Process Model	Mismatch	Figure No.	M_p SP	Time-Domain Response M_p	t_r	t_s	Comments
PM3	P=3 M=2 C=1	7.13	15%	0%	75.0	117.0	Time-domain performance improves as M_pSP tuning knob is increased.
			20%	0%	68.0	108.0	
			25%	0%	50.3	84.0	
			30%	0%	27.4	72.0	
PM4	P=3 M=2 C=1	7.14	15%	0%	30.3	46.0	Rise time improves as M_pSP tuning knob is increased. Settling time does the same until an overshoot is observed.
			20%	0%	21.9	40.0	
			25%	0%	15.8	19.3	
			30%	4%	12.4	35.0	

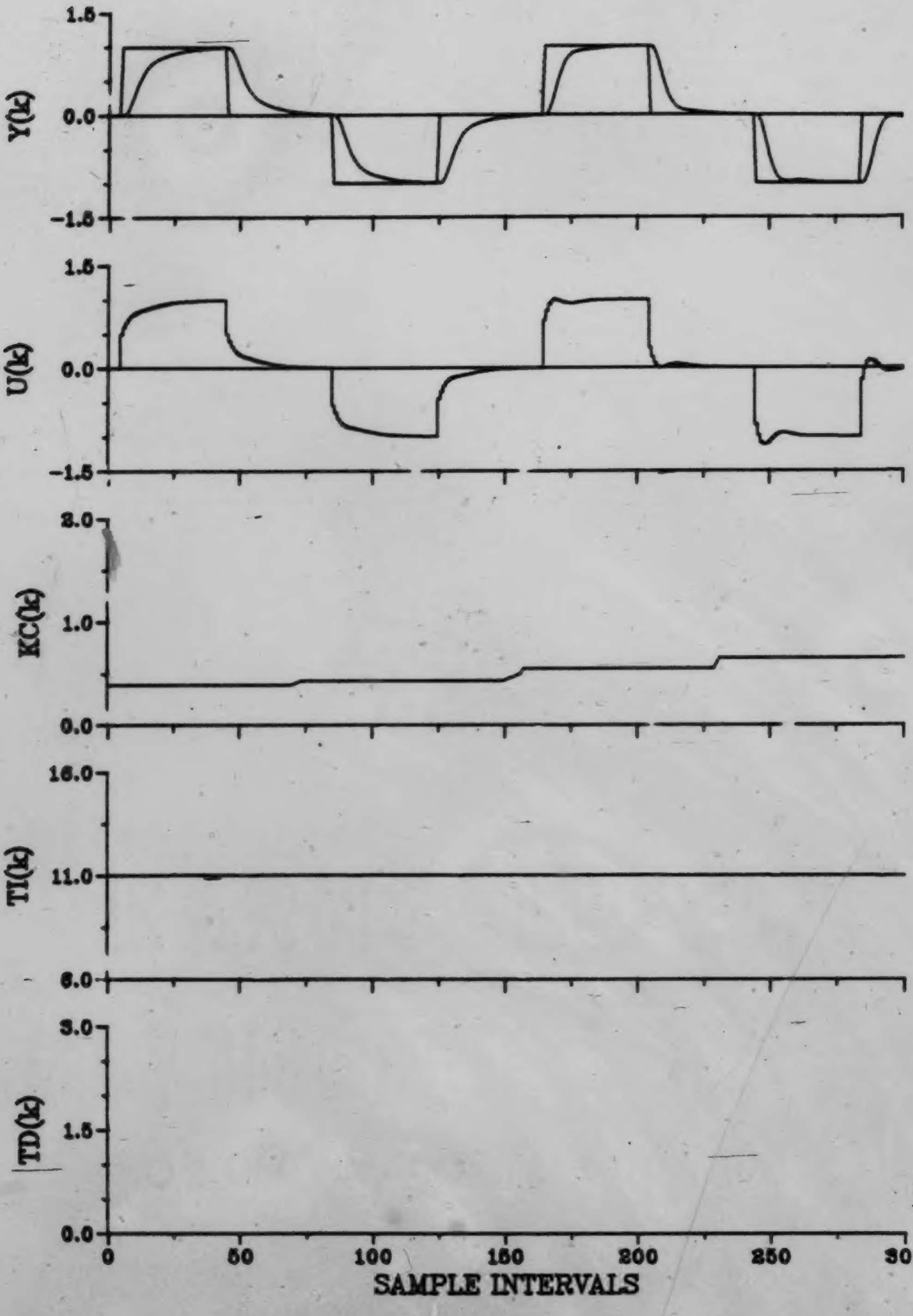

Figure 7.13 Online Mp setpoint (15-30%) tuning of self-tuning PI controller for Process Model 3

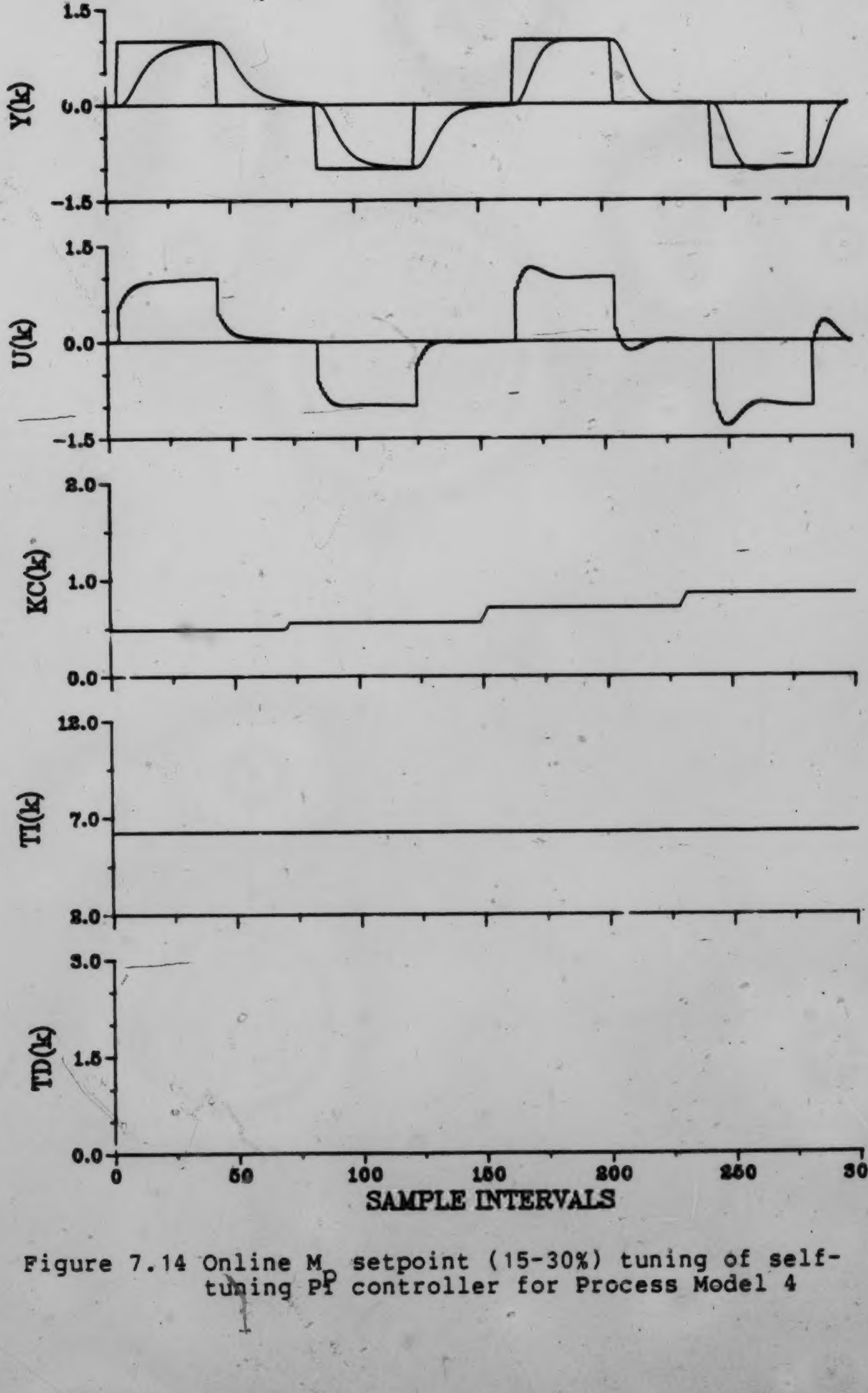

Figure 7.14 Online M_p setpoint (15-30%) tuning of self-tuning PI controller for Process Model 4

found to have difficulties. The calculated controllers give unstable closed-loop results. This is attributed to the fact that the first order model gives insufficient information about the process to carry out a reasonable controller design.

7.4.2 PID Control and Mismatch

The performance of the self-tuning PPPID control for mismatch is summarized in Table 7.4. Five of the linear processes listed in Table 5.1 result in a model order mismatch when controlled by a second order PID controller. These processes are either first or third order. Figures 7.15 - 7.18 show the self-tuning runs for four of these cases. Each run begins with the initial parameters listed in Table 7.1.

The self-tuning PID algorithm controls PM1 and PM6 very well. Since PM1 is first order and the controller second order, the order mismatch involves parameter estimation with an overparameterized model. Figure 7.15 shows a fast overdamped response with moderate control action. Estimated process parameter convergence is excellent with the second a_i and b_i parameters having values close to zero but not identically zero. Figure 7.18 shows the control performance for a third order process with numerator dynamics (PM6). Control produces a fast overdamped response with moderate control action.

Figures 7.16 and 7.17 summarize the closed-loop

Table 7.4 Performance of self-tuning PID controller in the presence of process/model order mismatch

Process Model	Mismatch	Figure No.	M_p SP	Time-Domain Response			Comments
				M_p	t_r	t_s	
PM1	P=1 M=2 C=2	7.15	15%	0%	4.9	7.7	Control action moderate, fast overdamped response
PM3	P=3 M=2 C=2	7.16	15%	0%	66.0	96.0	Control action sluggish, heavily overdamped response. Increase M_pSP
PM4	P=3 M=2 C=2	7.17	15%	0%	199.3	297.0	Response is so slow that the system is essentially open-loop. Increase M_pSP
PM6	P=3 M=2 C=2	7.18	15%	0%	28.5	34.5	Control action moderate, fast overdamped response

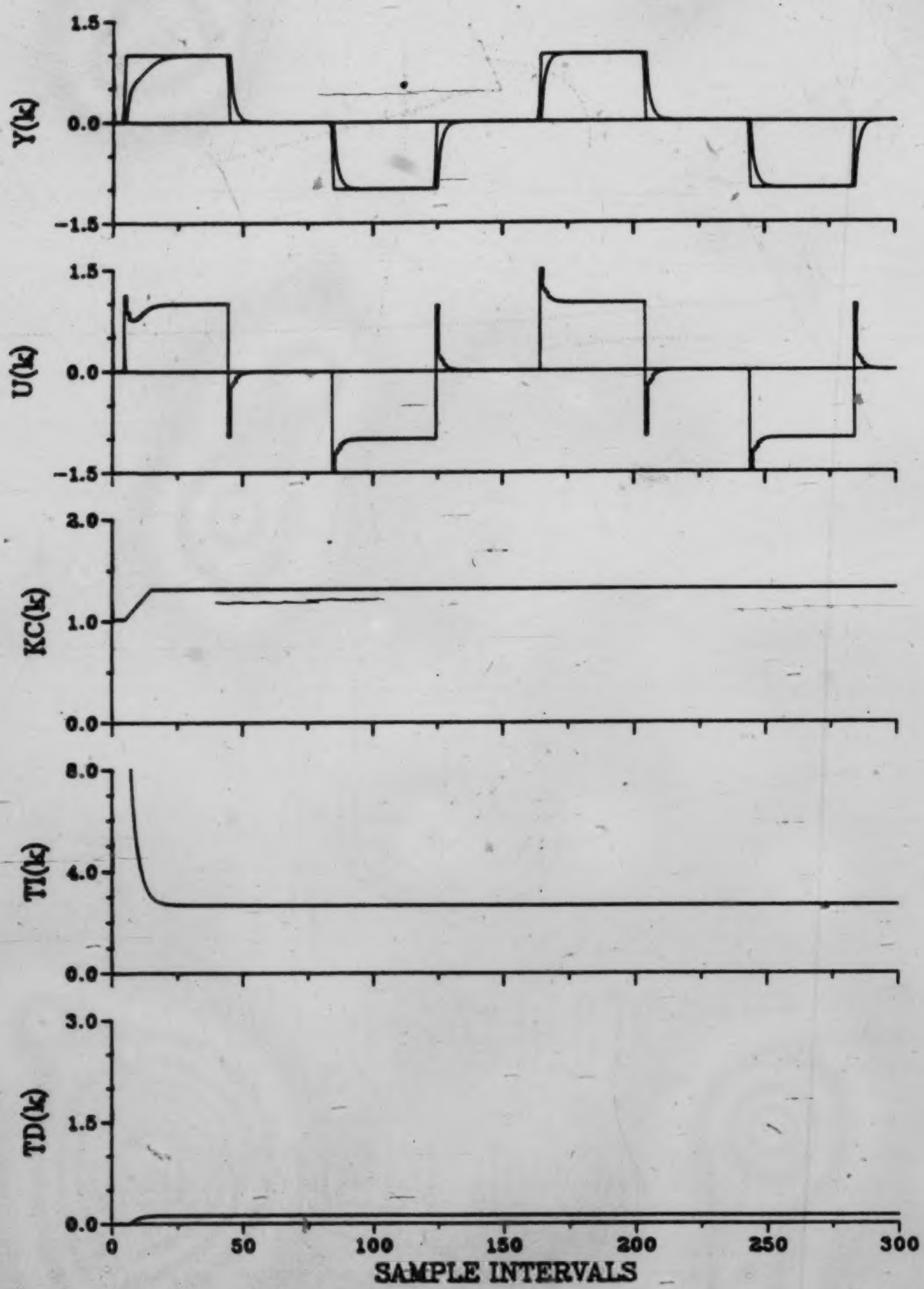

Figure 7.15 Servo self-tuning PID control of Process Model 1 (INC,P=1,M=C=1,$M_p$sp=15)

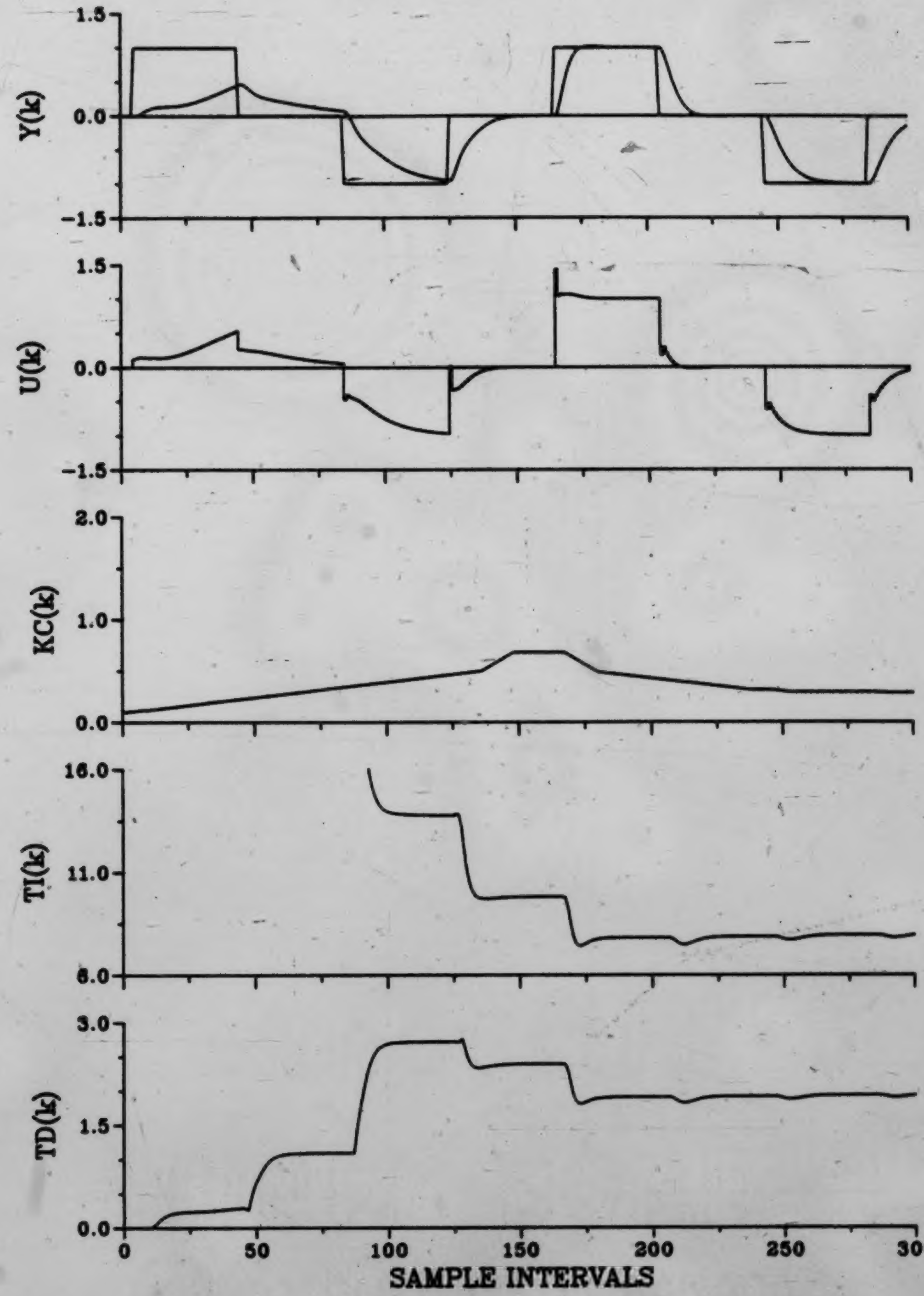

Figure 7.16 Servo self-tuning PID control of Process Model 3 (INC,P=3,M=C=2,$M_p$sp=15)

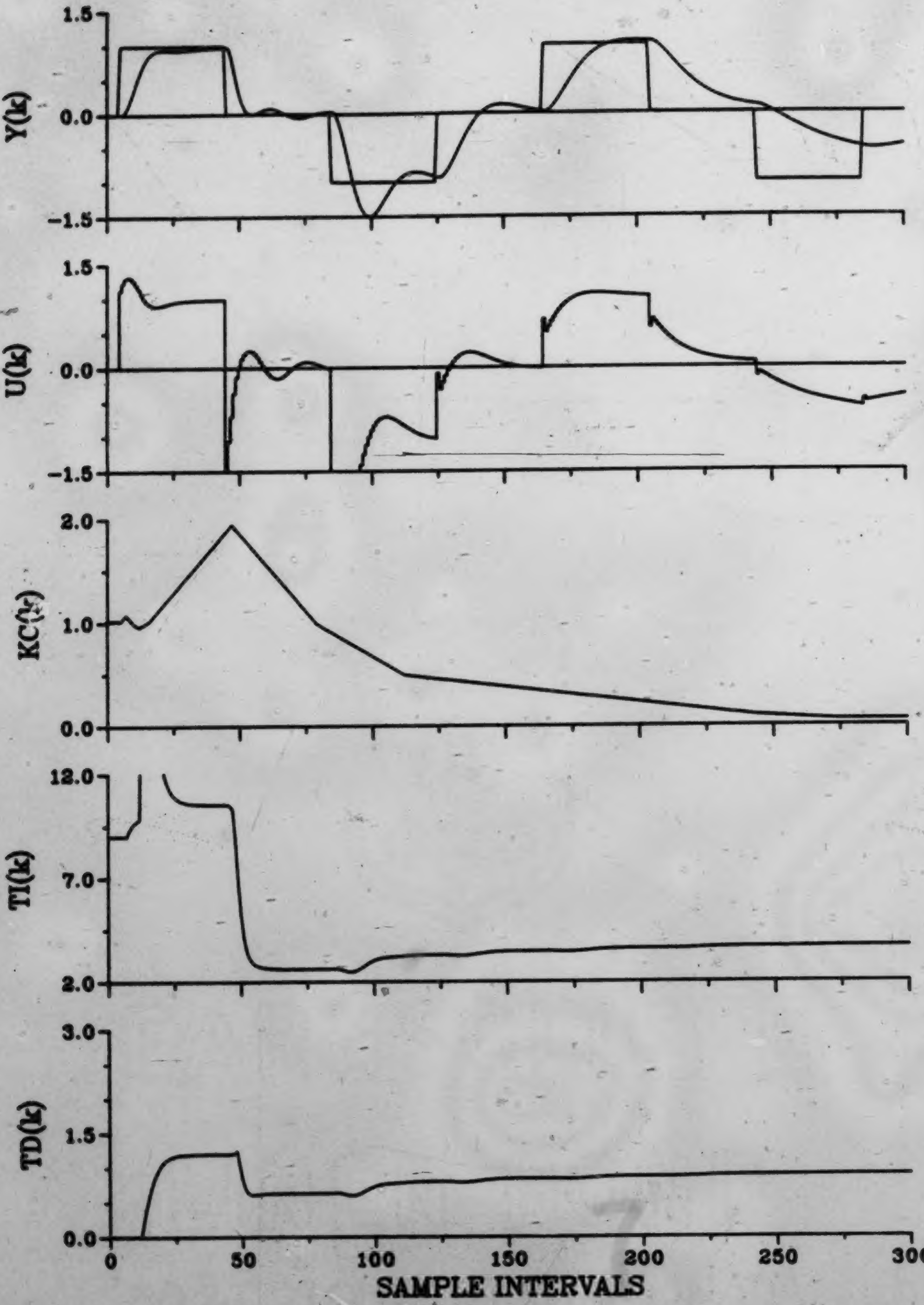

Figure 7.17a Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=15)

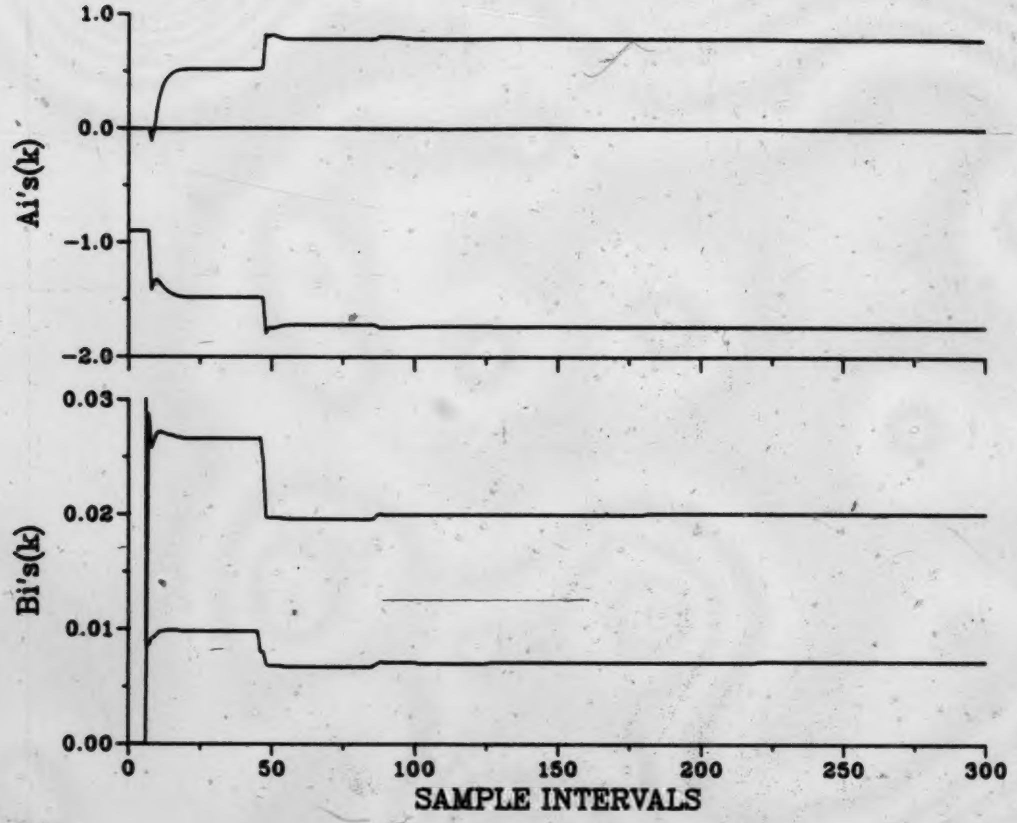

Figure 7.17b Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=15)

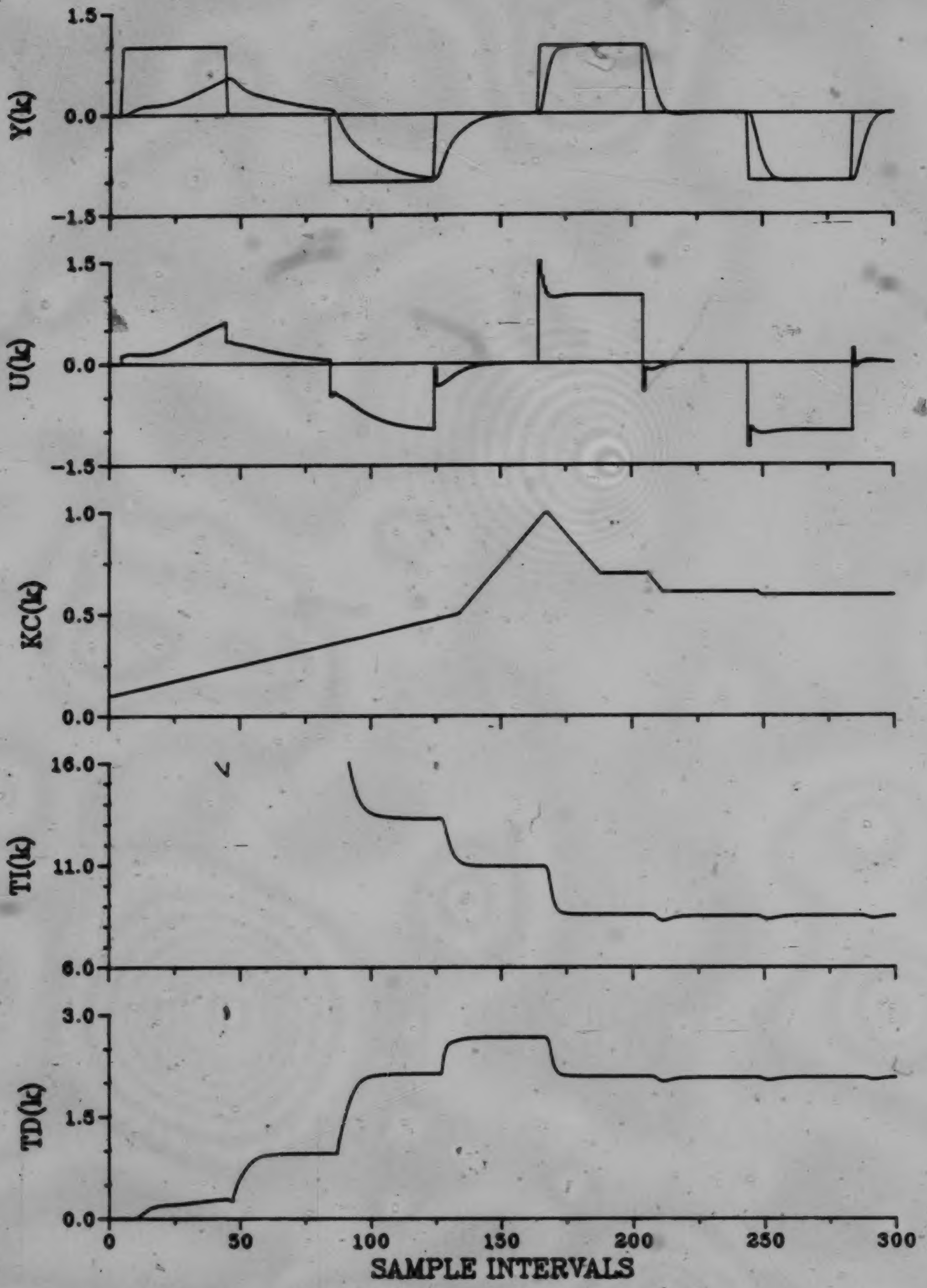

Figure 7.18 Servo self-tuning PID control of Process Model 6 (INC,P=3,M=C=2,$M_p$sp=15)

performance for two third order processes with no numerator dynamics (PM3 and PM4). Both responses are heavily overdamped. The response of PM4 for a M_p setpoint of 15% is very poor with rise and settling times of 199.3 and 297.0 respectively. For PID control PM4 represents the worst case for order mismatch. The process is third order and sampled at a rate where the high frequency components of the process are significant.

As mentioned previously, in the presence of mismatch the M_p setpoint becomes a tuning knob to shape the closed-loop response of the system. This is illustrated in Table 7.5 for PM3 and PM4. The M_p setpoint is increased from 15-30% in 5% increments. The simulation runs are seen in Figures 7.19 and 7.20. Converged process parameter estimates are used to initialize these runs. A marked improvement in closed-loop response is seen as the M_p setpoint is increased. The rise and settling times decreased until the response becomes underdamped. Settling times began to increase when the peak overshoot is greater than 1%. Process Model 4 is characterized by long settling times for PID control. This is explained by examination of the Figure 6.21 where the root locus plot for this case is shown. The root loci lie close to the unit circle near z=1. This area is characterized by long rise and settling times. Improved performance could be achieved if the sampling period is longer as this would move the process poles toward the origin of the unit circle. This would move

Table 7.5 Online tuning of M_p setpoint for self-tuning PID control

Process Model	Mismatch	Figure No.	M_p SP	Time-Domain Response			Comments
				M_p	t_r	t_s	
PM3	P=3 M=2 C=2	7.19	15%	0%	66.0	96.0	Rise time improves as M_pSP tuning knob is increased. Settling time does the same until an overshoot is observed.
			20%	0%	51.8	72.0	
			25%	1%	35.0	42.0	
			30%	3%	26.7	58.0	
PM4	P=3 M=2 C=2	7.20	15%	0%	199.3	297.0	Rise time improves as M_pSP tuning knob is increased. Settling time does the same until an overshoot is observed.
			20%	0%	96.4	144.7	
			25%	1%	30.0	35.9	
			30%	6%	18.7	44.1	

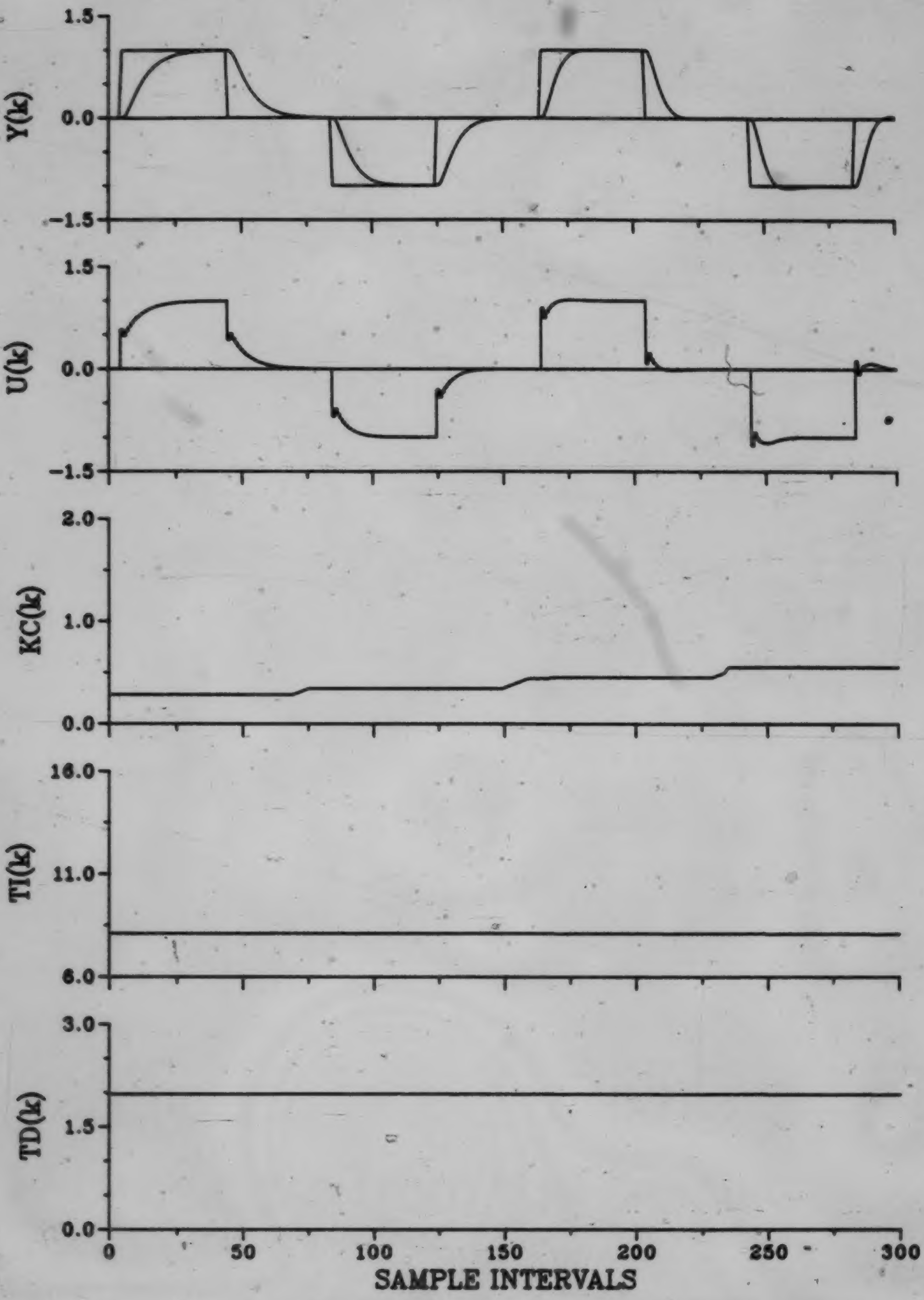

Figure 7.19 Online Mp setpoint (15-30%) tuning of self-tuning PID controller for Process Model 3

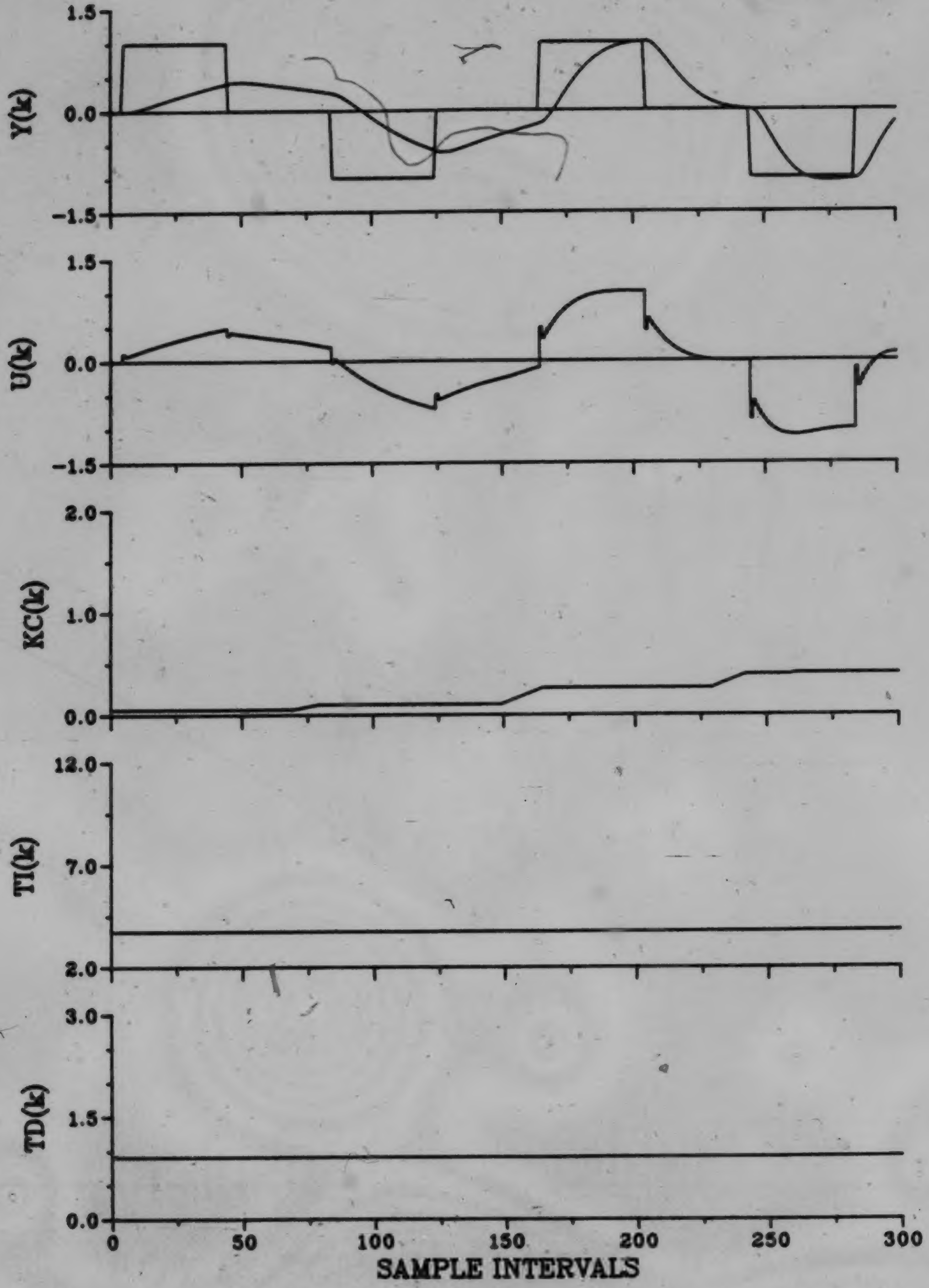

Figure 7.20 Online Mp setpoint (15-30%) tuning of self-tuning PID controller for Process Model 4

the root loci further into the unit circle where better time-domain response is achieved. In addition sampling at a slower rate would improve the estimated second order model as the high frequency portion of PM4 would be attenuated.

The control performance of the self-tuning PID controller with Process Model 4 is improved by increasing the M_p setpoint or sampling at a slower rate. It could also be improved by moving the controller zeroes. The controller zeroes are determined by T_i and T_d which are in turn determined by the a_i parameters of the estimated second order model with the restriction that T_d must be less than $T_i/4.1$ for an overdamped system. The effect of the estimated model is examined for PM3 and PM4 by comparing the results of incremental and positional variable identification. The results of this study are shown in Table 7.6 for PM4 with a M_p setpoint of 15% and 30%. The simulation results for this study are shown in Figures 7.17 and 7.21-7.23. For this third order process the improvement in control is considerable by the use of positional variable identification. The results of Figure 7.22 show rise and settling times one tenth those of the equivalent incremental variable case. If the M_p setpoint is increased to 30%, the response becomes underdamped with good rise and settling times.

It can be concluded that positional variable identification results in better PID control of PM4. The reason for this can be explained by examining the second

Table 7.6 Effect of identification method on self-tuning PID control in the presence of process/model mismatch

Process Model	Mismatch	Figure No.	M_p SP	Time-Domain Response M_p	t_r	t_s	ID Method	Comments
PM4	P=3	7.17	15%	0%	199.3	297.0	INC	System response extremely slow
	M=2							
	C=2	7.21	30%	6%	18.7	44.1	INC	Sluggish underdamped response
		7.22	15%	0%	16.8	22.5	POS	Reasonable response
		7.23	30%	18%	5.0	19.3	POS	Underdamped response

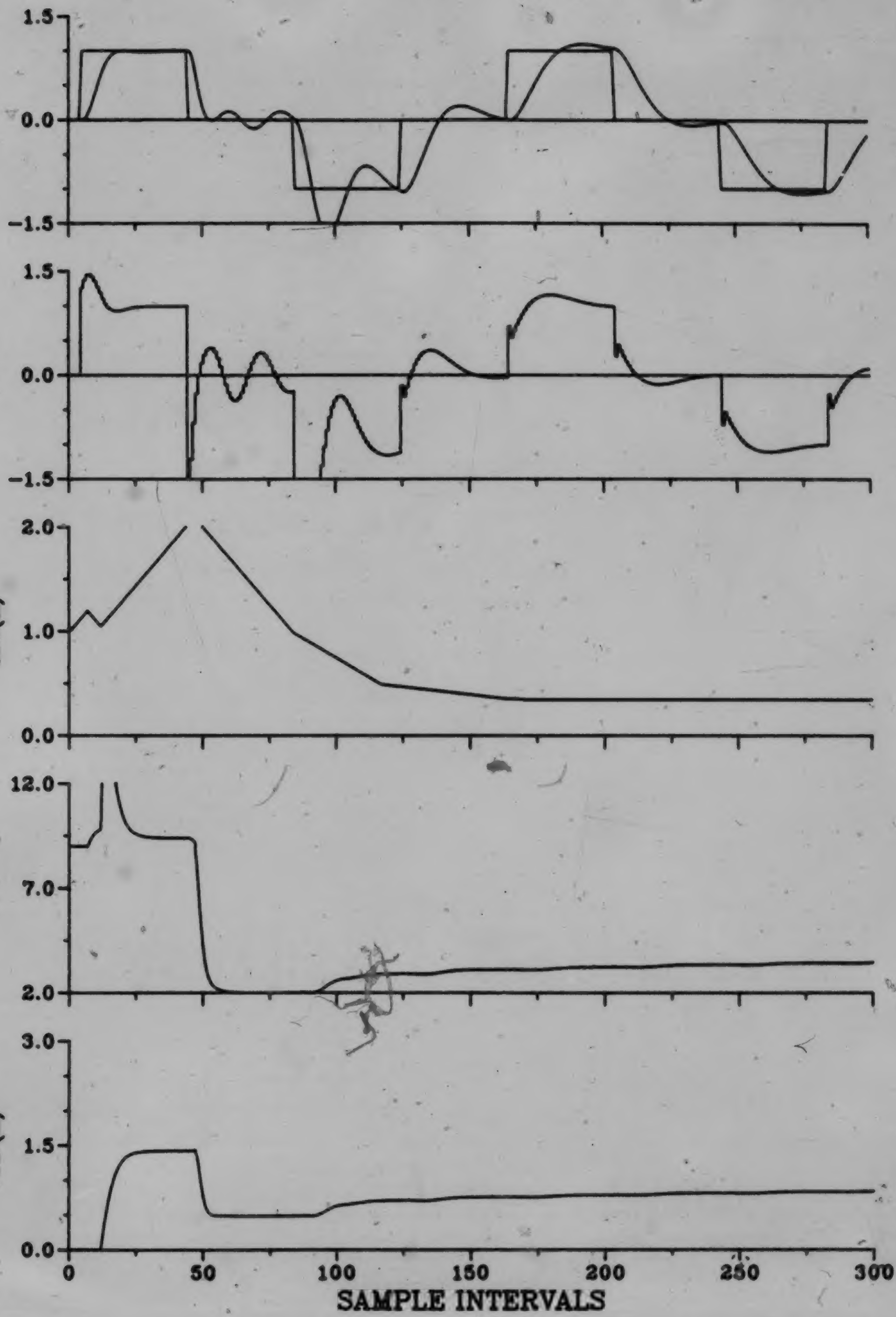

Figure 7.21a Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=30%)

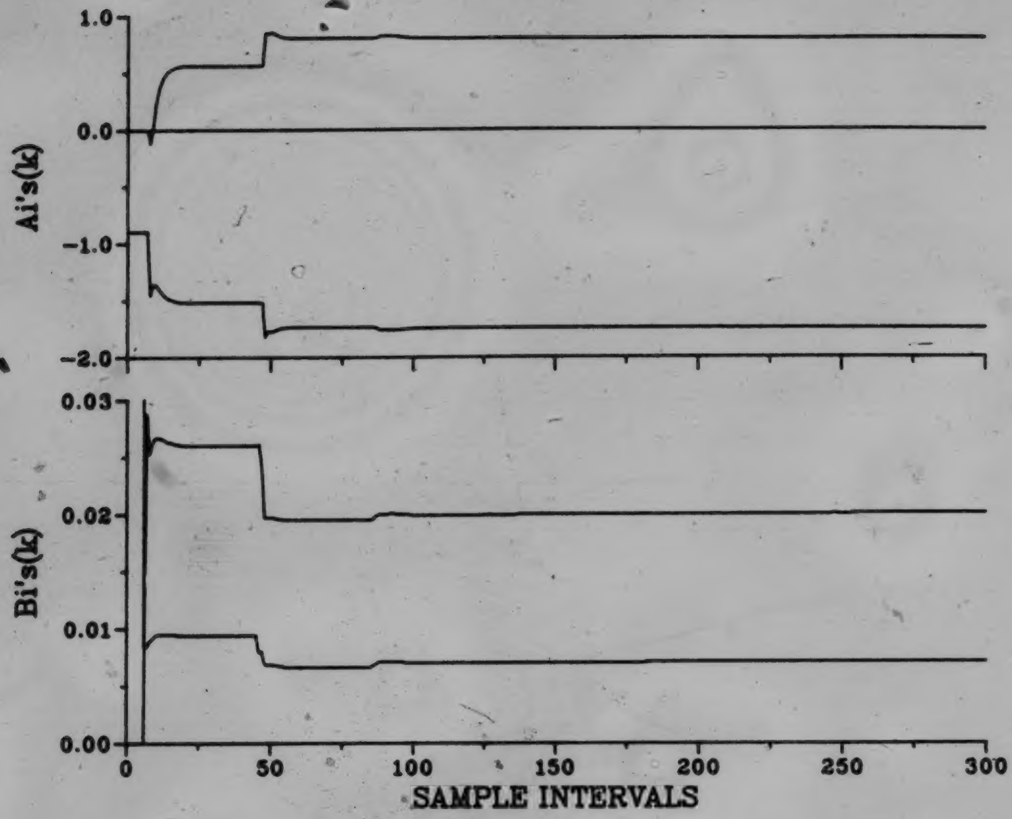

Figure 7.21b Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=30)

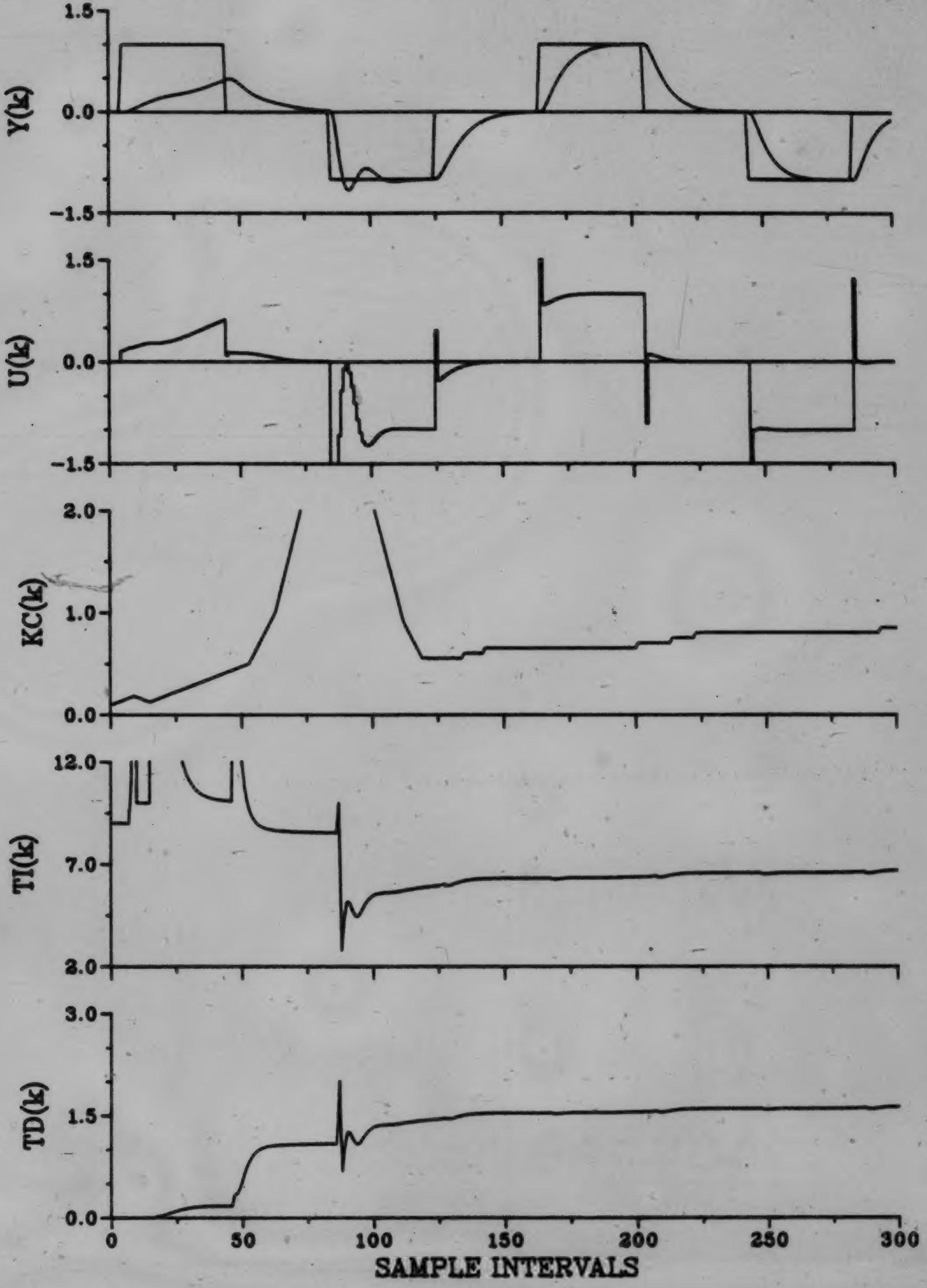

Figure 7.22a Servo self-tuning PID control of Process Model 4 (POS,P=3,M=C=2,$M_p$sp=15)

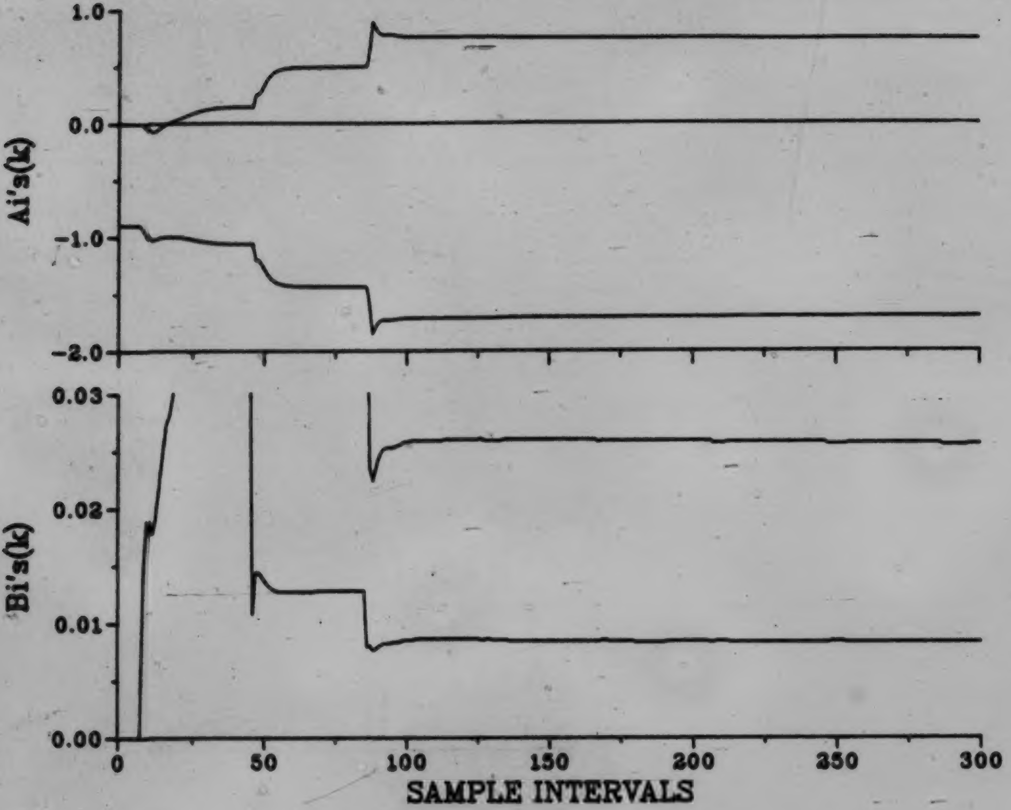

Figure 7.22b Servo self-tuning PID control of Process Model 4 (POS,P=3,M=C=2,$M_p$sp=15)

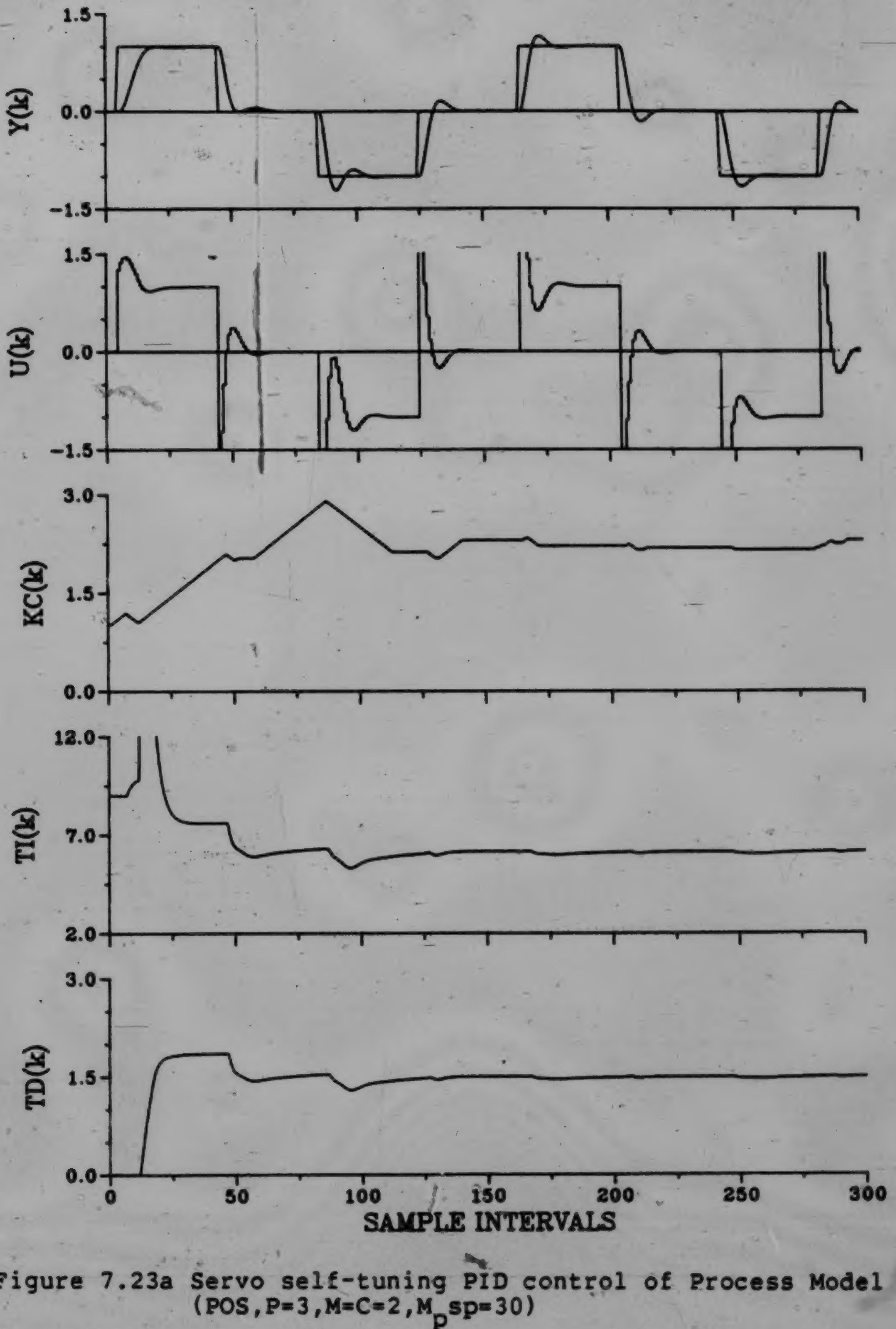

Figure 7.23a Servo self-tuning PID control of Process Model 4
(POS, $P=3, M=C=2, M_p sp=30$)

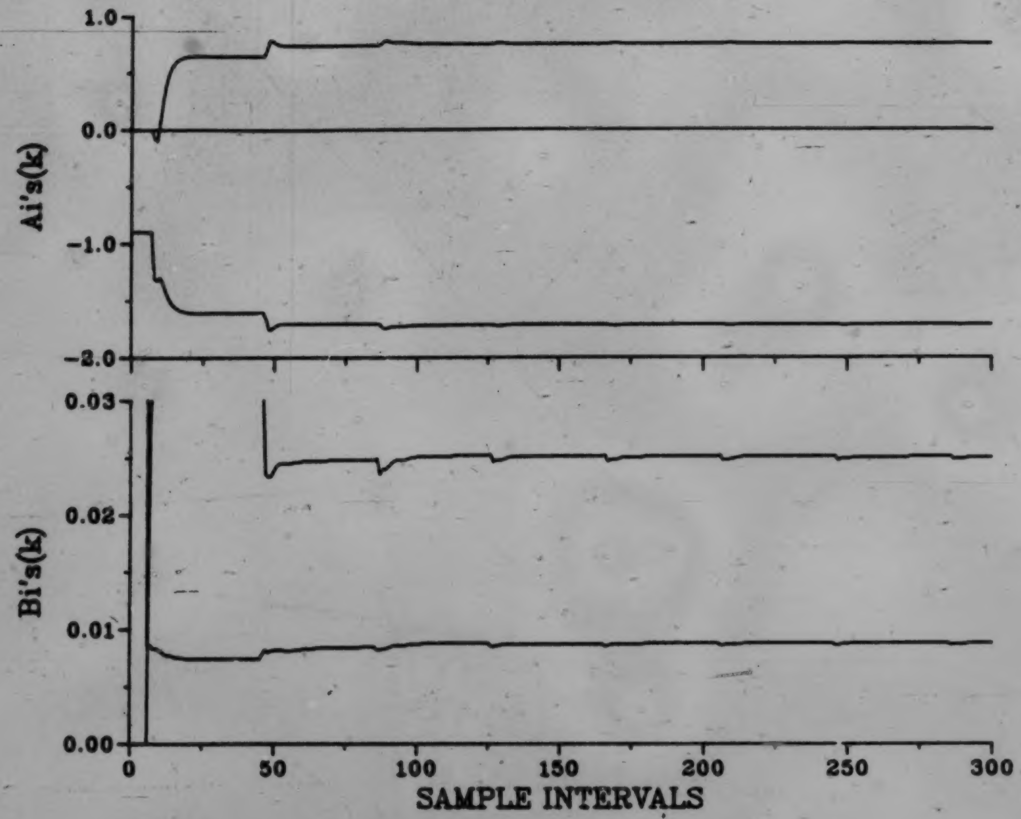

Figure 7.23b Servo self-tuning PID control of Process Model 4 (POS,P=3,M=C=2,$M_p$sp=30)

order model which each variable type estimates. If there is no mismatch between the process order and the estimated model order, both incremental and positional variables will result in the true parameter values assuming no noise and zero-mean data. In the presence of process/model order mismatch, there are no true parameter estimates. One can only speak of a converged set of parameter estimates with incremental and positional variable resulting in a different set of converged parameter estimates. This is shown in Table 7.7 for PM3 and PM4. Although both these third order processes are overdamped, the estimated second order models are underdamped with each identification method having a different parameter estimate set. The open-loop step response of these estimated models is compared with the actual process in Figures 7.24 and 7.25. The estimated models from positional variable identification match the process response over the whole dynamic range and in the steady state. The gain of the positional variable models is close to 1.0 while for PM4 the incremental variable model has a gain of 0.59. The incremental variable estimation models match the process response mainly in the initial dynamic portion. The b_0 parameters of the incremental models are very close to the b_0 parameters of the actual processes. This is due to the incremental model matching the high frequency part of the process which is more predominant in Process Model 4. This preference for the high frequency dynamics is a result of differencing the

Table 7.7 Effect of identification method on estimated model in the presence of process/model mismatch

Process Model	Mismatch	Figure No.	ID Method	Estimated Process Parameters					Comments
				a_1	a_2	b_0	b_1	Gain	
PM3	P=3 M=2	7.24	INC	-1.527	.6273	.0265	.0619	0.88	Model is underdamped with poles at 0.763 + j0.211, ζ=0.65
			POS	-1.458	.5568	.0278	.0700	0.99	Model is underdamped with poles at 0.729 + j0.158, ζ=0.81
PM4	P=3 M=2	7.25	INC	-1.736	.7824	.0072	.0201	0.59	Model is underdamped with poles at 0.868 + j0.169, ζ=0.54
			POS	-1.697	.7314	.0084	.0252	0.98	Model is underdamped with poles at 0.849 + j0.107, ζ=0.78

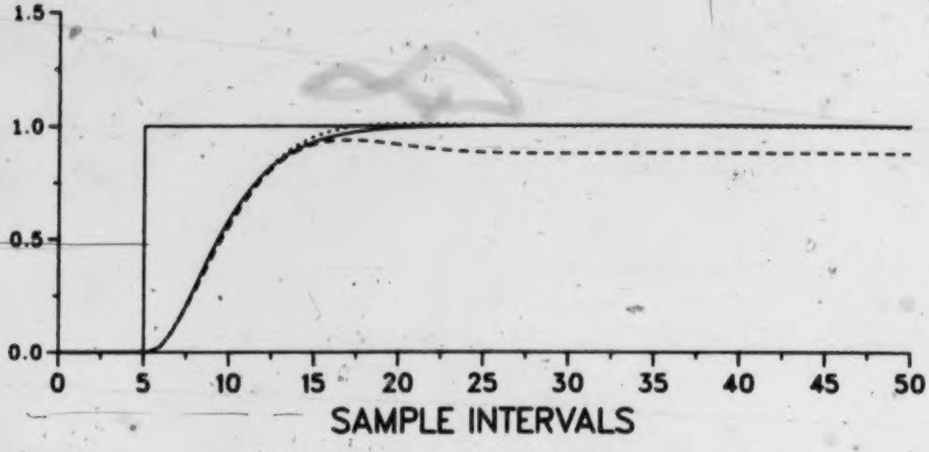

Figure 7.24 Open-loop response of Process Model 3 (——) and estimated second order models from incremental (----) and positional (····) identification

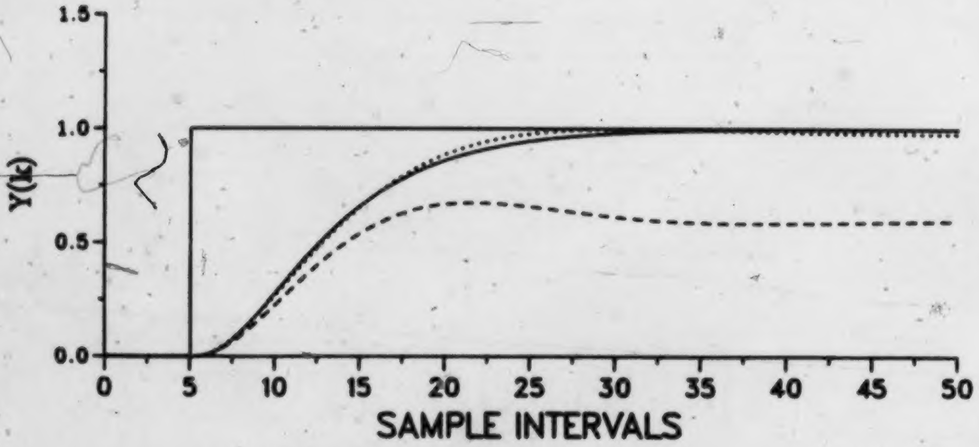

Figure 7.25 Open-loop response of Process Model 4 (——) and estimated second order models from incremental (----) and positional (····) identification

data. The differencing operator acts as a high pass filter.

This difference in estimated parameters for incremental and positional variable identification methods results in different controller zero locations. The positional model places the controller zeroes closer to (1,j0) in the unit circle of the z-plane. This causes the root loci to branch between the process pol[illegible] well within the unit circle and results in better rise and settling times. The increment model places the controller zeroes among the process poles. This causes the root loci to branch between the largest process pole and the controller pole at (1,j0). This results in poor rise and settling times.

In summary the self-tuning PPPID controller performs well in the presence of process/model order mismatch. The M_p setpoint can be used to speed the response of the closed-loop system if required. The controller performance for the worst mismatch cases (PM3, PM4) can be improved by model parameter estimation using positional variables. This is found to locate the controller zeroes to a location which gives improved closed-loop response. However, the use of positional variables introduces other problems especially for non-zero mean data.

7.5 Summary

The modified self-tuning PPPI(D) control law proposed in Chapter 6 has been evaluated with and without the presence of process/model order mismatch. As a result of these simulation studies it can be concluded that this self-tuning PI(D) controller is a reasonable controller design strategy for the calculation of controller constants (K_c, T_i, T_d). The positioning of the controller zeroes by T_i and T_d as seen in the complex z-plane results in good root loci location for time domain characteristics. The calculation of the controller gain (K_c) by solving the estimated characteristic equation of the closed-loop system and matching a damping ratio setpoint has been shown to be workable. The specific results of these simulation studies are summarized in point form below:

1. Control with no mismatch between the process and the estimated model resulted in excellent closed-loop performance. The maximum overshoot is as specified for processes with no numerator dynamics. In the presence of a mismatch between the process and estimated model order, control is good with the M_p setpoint becoming a tuning knob to shape the closed-loop response.
2. Incremental and positional variable identification results in a considerably different estimated model when a mismatch between process and model order exists. Incremental estimation tends to match the initial transients which are dominated by the higher frequency dynamics of the process. The incremental model does not match the steady state gain of the process. The positional model matches the process in the steady state and resulted in a better controller

design. Both methods estimated an underdamped model for the higher order over-damped process.

8. Performance of Self-Tuning PID Algorithm with Linear Systems: In the Presence of Delays and Disturbances

8.1 Introduction

The analysis of this chapter continues the studies begun in Chapter 7 for linear systems. The basis for these results are the same as for those presented in Chapter 7. In this chapter the objective is to evaluate the performance of this self-tuning PPPI(D) controller. The evaluation tests for the self-tuning controller focuses on three linear SISO process models outlined in Chapter 5 - PM1, PM2 and PM4. Performance in the presence of time delays, process bias and deterministic and stochastic disturbances is considered.

The layout of the chapter is as follows. Section 8.2 discusses the results of the self-tuning controller in the presence of time delays. Simulation results for deterministic disturbances are presented in Section 8.3. Section 8.4 considers the effect of stochastic disturbances on controller performance and parameter estimation, followed by a summary and comments in Section 8.5.

8.2 Simulation Results for Processes with Time Delays

Time delays are very common in the process industries and their presence complicates the use of advanced control strategies. From the frequency domain perspective time delays introduce additional phase lag and thus reduce the stability margin. From a z-domain view they introduce open-loop poles at the origin which cause the ultimate gain of the closed-loop system to be lowered significantly.

Time delays can be handled by a variety of methods. The controller can be detuned to allow closed-loop stability in the presence of time delays. Another approach is time delay compensation which predicts the undelayed output of the process and feedbacks this signal for control output calculation. This section will investigate these two methods for processes with time delays. The objective is to evaluate qualitatively the performance of the self-tuning PPPI(D) control law in the presence of time delays. Ideal and model-plant mismatch cases for process/model order and time delay estimate are considered.

8.2.1 Controller Detuning

The simulation results for this method of handling time delays are summarized in Table 8.1. The two processes considered are PM2 and PM4 for the ideal case as well as a mismatch in the time delay estimate. In addition the performance of the self-tuning PPPID in the presence of a large time delay is considered.

Figure 8.1 shows the results for the ideal case of no mismatch in the process/model order and time delay estimate. The time delay is unity (i.e. d=1 or k=d+1) and known. Control performance is excellent once process

Table 8.1 Performance of self-tuning PID control in the presence of time delays

Process Model	Mismatch Order	Mismatch Delay	Figure No.	M_p SP	Time-Domain Response M_p	Time-Domain Response t_r	Time-Domain Response t_s	Comments
PM2	P=2 M=2 C=2	da=1 de=1	8.1	30%	12%	4.9	15.8	Excellent underdamped response
		da=1 de=0	8.2	15%	3%	19.7	42.0	Slow underdamped response, large unstable oscillations during tuning
		da=0 de=1	8.3	20%	25%	3.0	16.8	Good underdamped response, large unstable oscillations during tuning
		da=4 de=4	8.4 8.5	40%	0%	18.7	22.0	Excellent overdamped response
PM4 (T_s=1)	P=3 M=2 C=2	da=1 de=1	8.6	15%	0%	30.4	39.0	Slow overdamped response, increase M_pSP
		da=1 de=0	8.7	15%	-	-	-	Response is essentially open-loop due to both order and delay mismatch.
		da=0 de=0-1	8.8	15%	5%	11.1	28.7	Reasonable underdamped response
		da=4 de=4	8.9	40%	27%	16.7	80.0	Slow underdamped response
PM4 (T_s=2)	P=3 M=2 C=2	da=4 de=4	8.10	40%	2%	40.1	75.0	Slow overdamped response

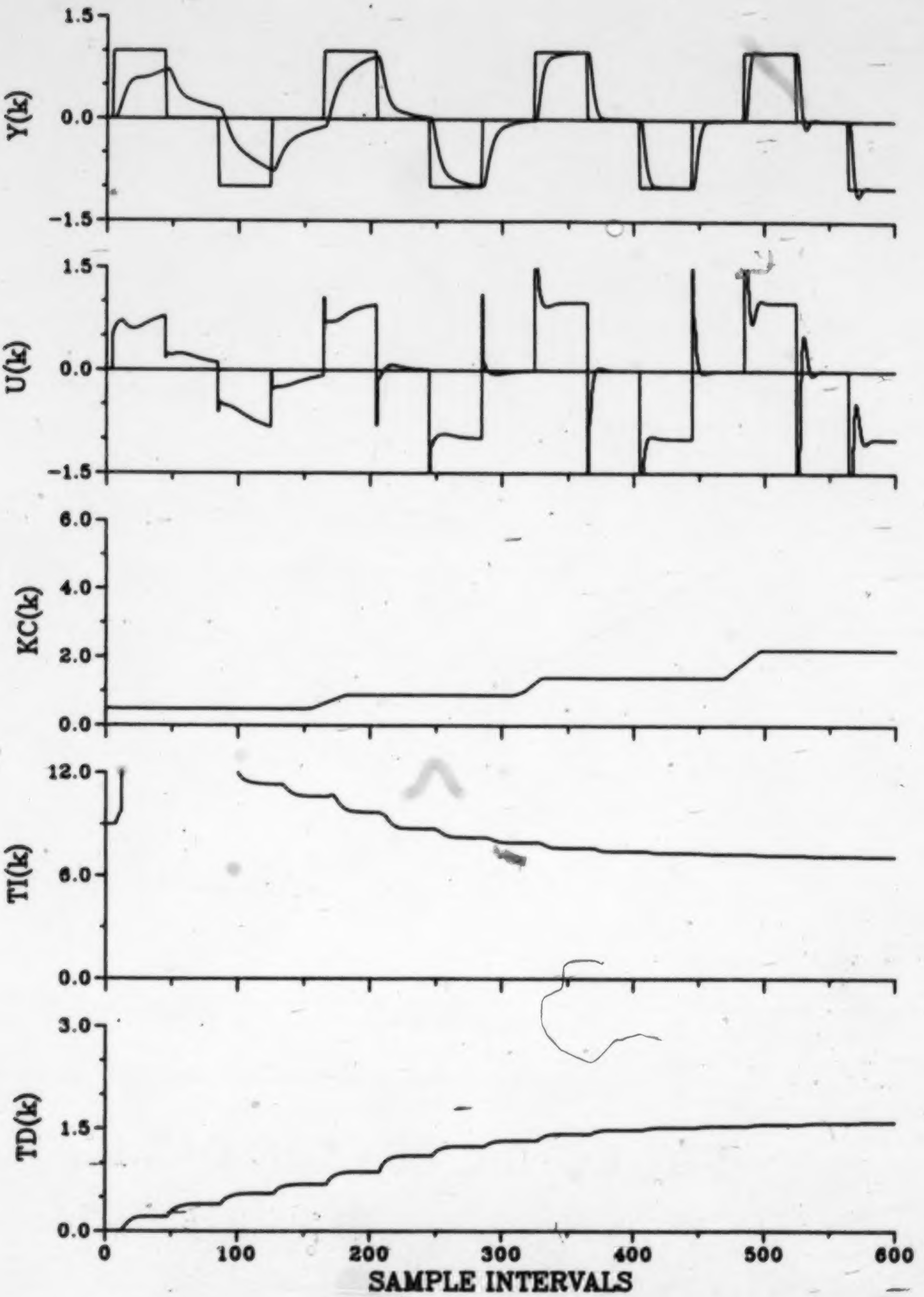

Figure 8.1 Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15-30,da=de=1)

parameter estimates have converged. Note that the peak overshoot setpoint is varied from Mp=15-30% in 5% increments. The actual M_p differs from the setpoint and the Mp setpoint has become a tuning knob. The controller gain calculation method can naturally take time delays into account since it solves the root loci ~~of~~ the estimated model with time delay and the self-tuning controller.

The effect of a mismatch in the time delay estimate is shown in Figure 8.2 and 8.3. In the first run the actual time delay is d=1 but is considered as d=0 in the estimation algorithm. Control performance is reasonable with long rise and settling times. In the second run the actual delay is d=0 but estimated to be d=1. The control performance is good once process parameter estimates have converged but is unstable during the tuning period. This was also observed in Figure 8.2 during the tuning period. These large oscillations are attributed to the mismatch of data in the input/output regressor and the sensitivity of incremental variable identification. This unacceptable tuning behaviour can be avoided if initial parameter estimation is done in the background or by estimating additional b_i parameters to span the estimated minimum and maximum time delay.

The performance of the self-tuning PID controller in the presence of a significant known time delay (i.e. d=4) is shown in Figure 8.4. The control law produces an excellent overdamped response. The time delay adds four open-loop poles at the origin of the unit circle as seen in Figure

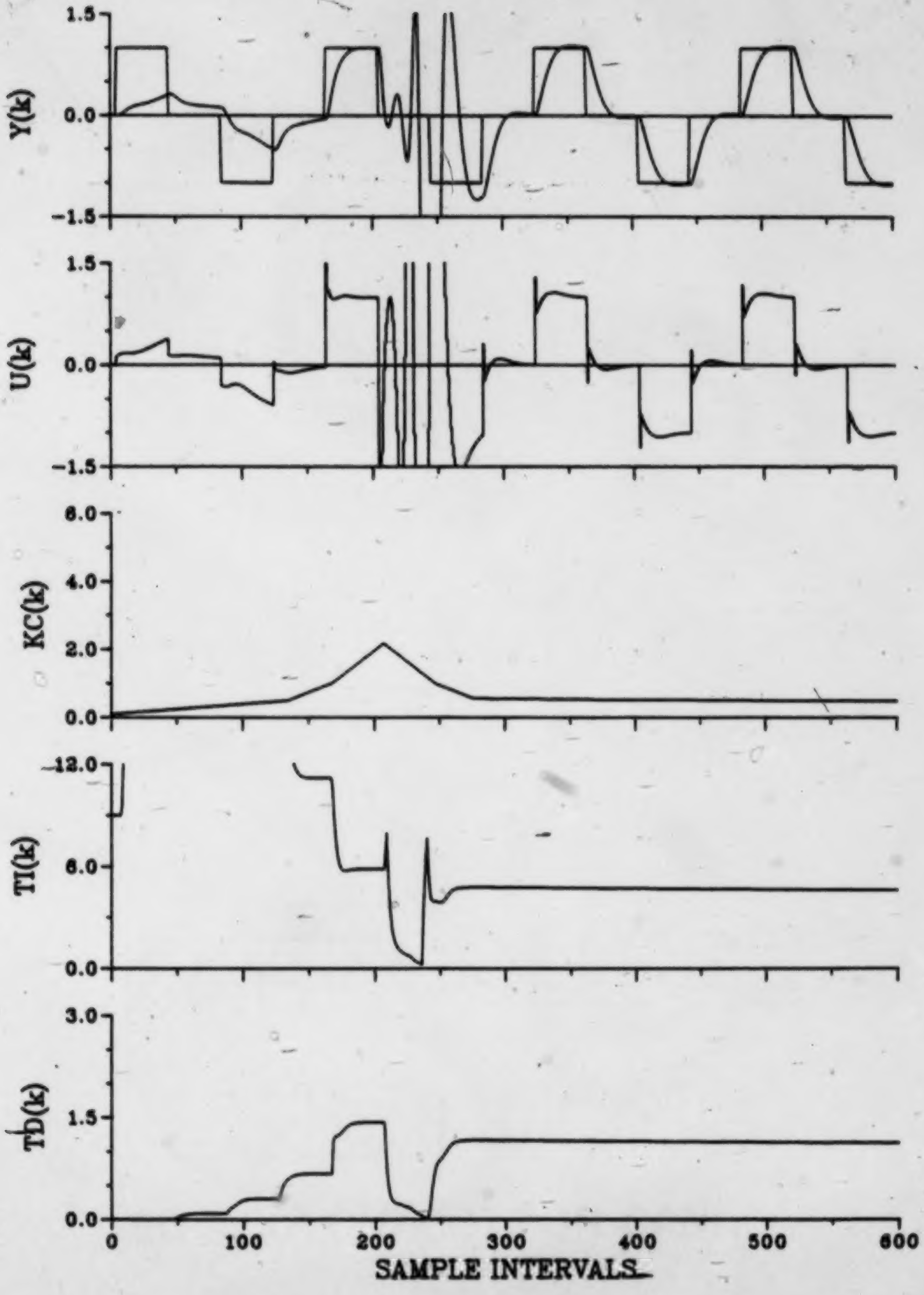

Figure 8.2a Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15,da=1,de=0)

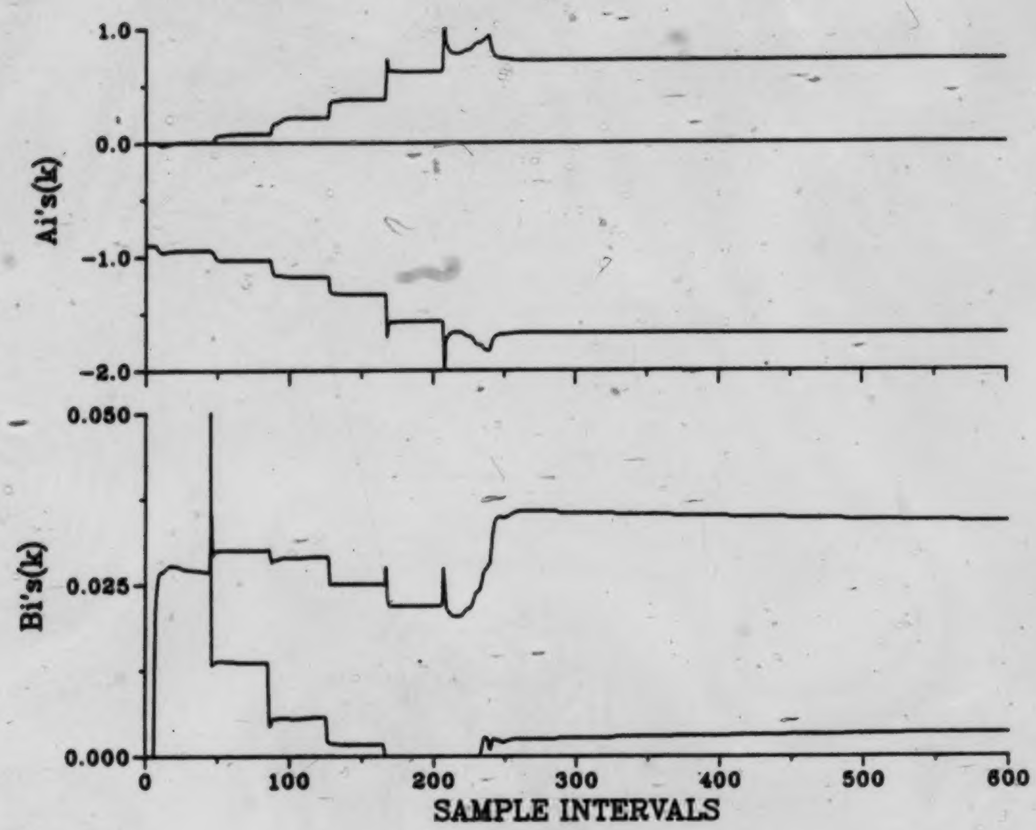

Figure 8.2b Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15,da=1,de=0)

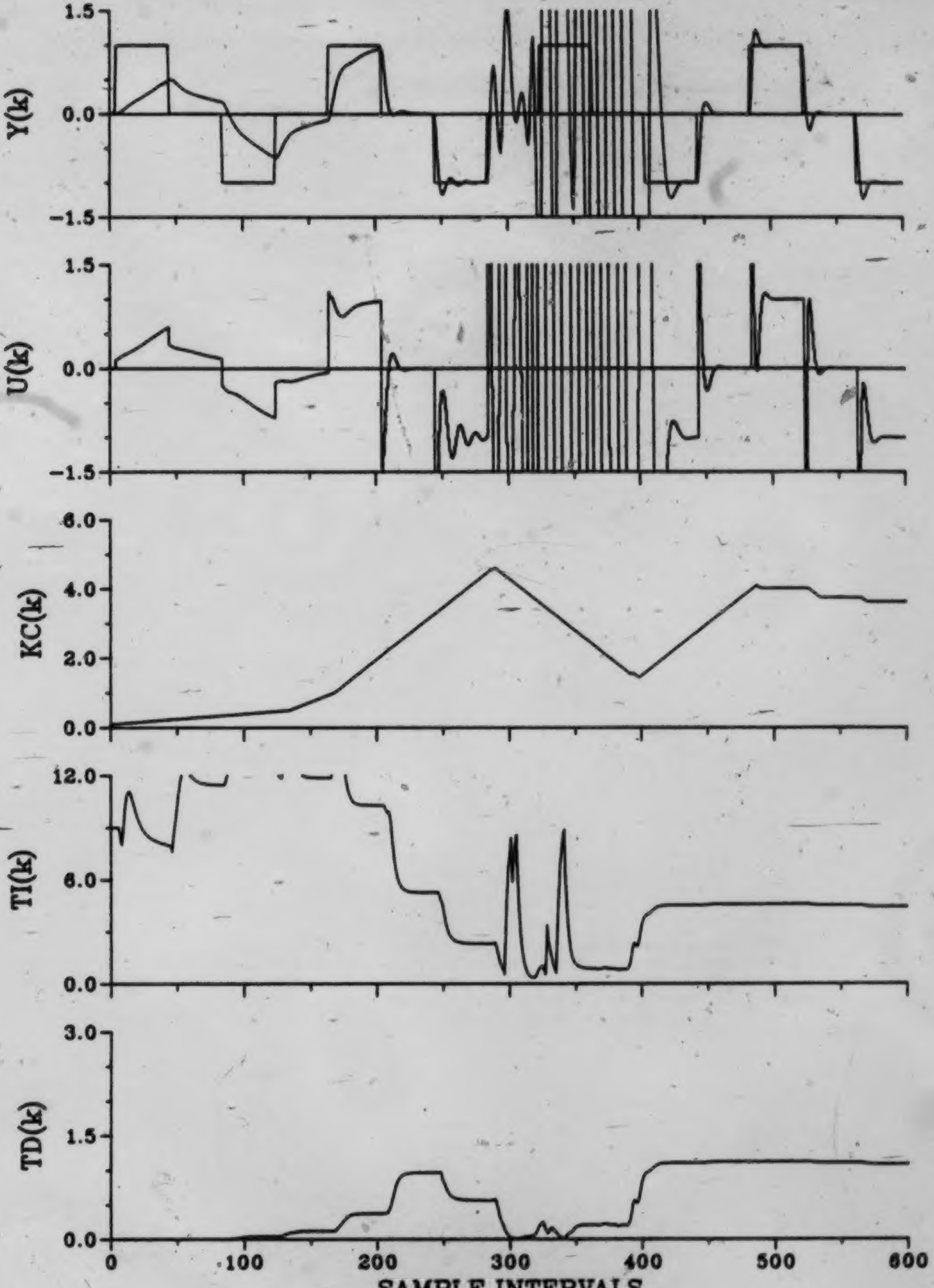

Figure 8.3a Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=20,da=0,de=1)

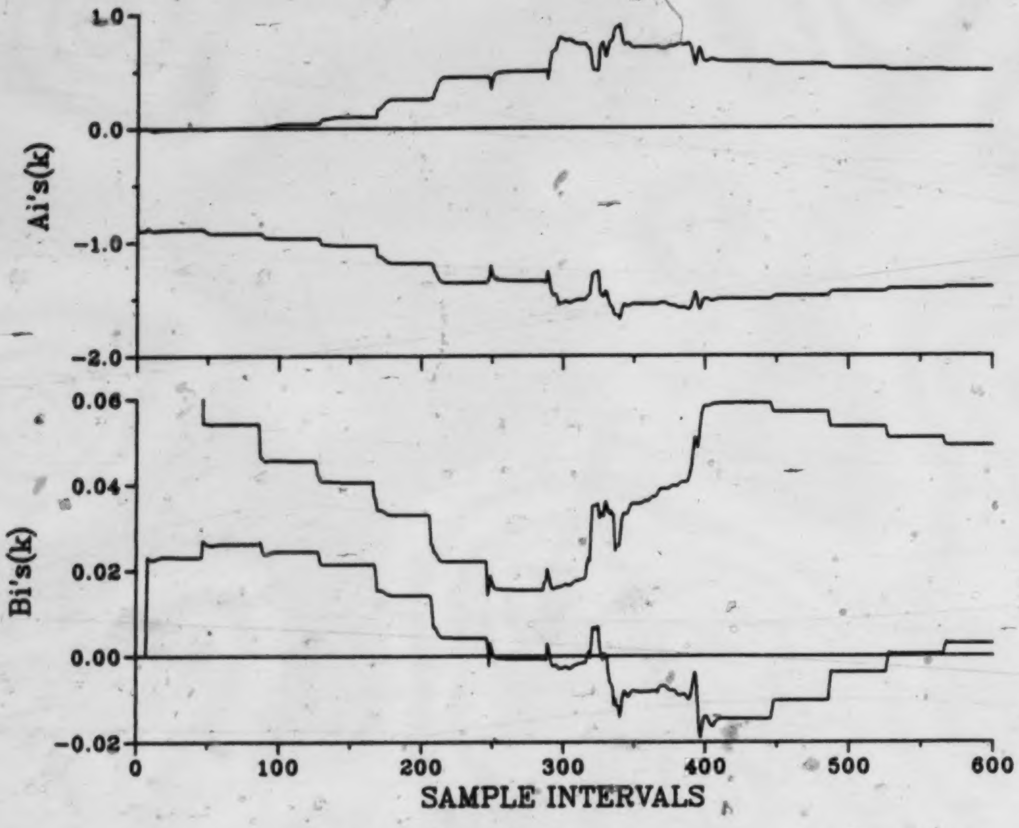

Figure 8.3b Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=20,da=0,de=1)

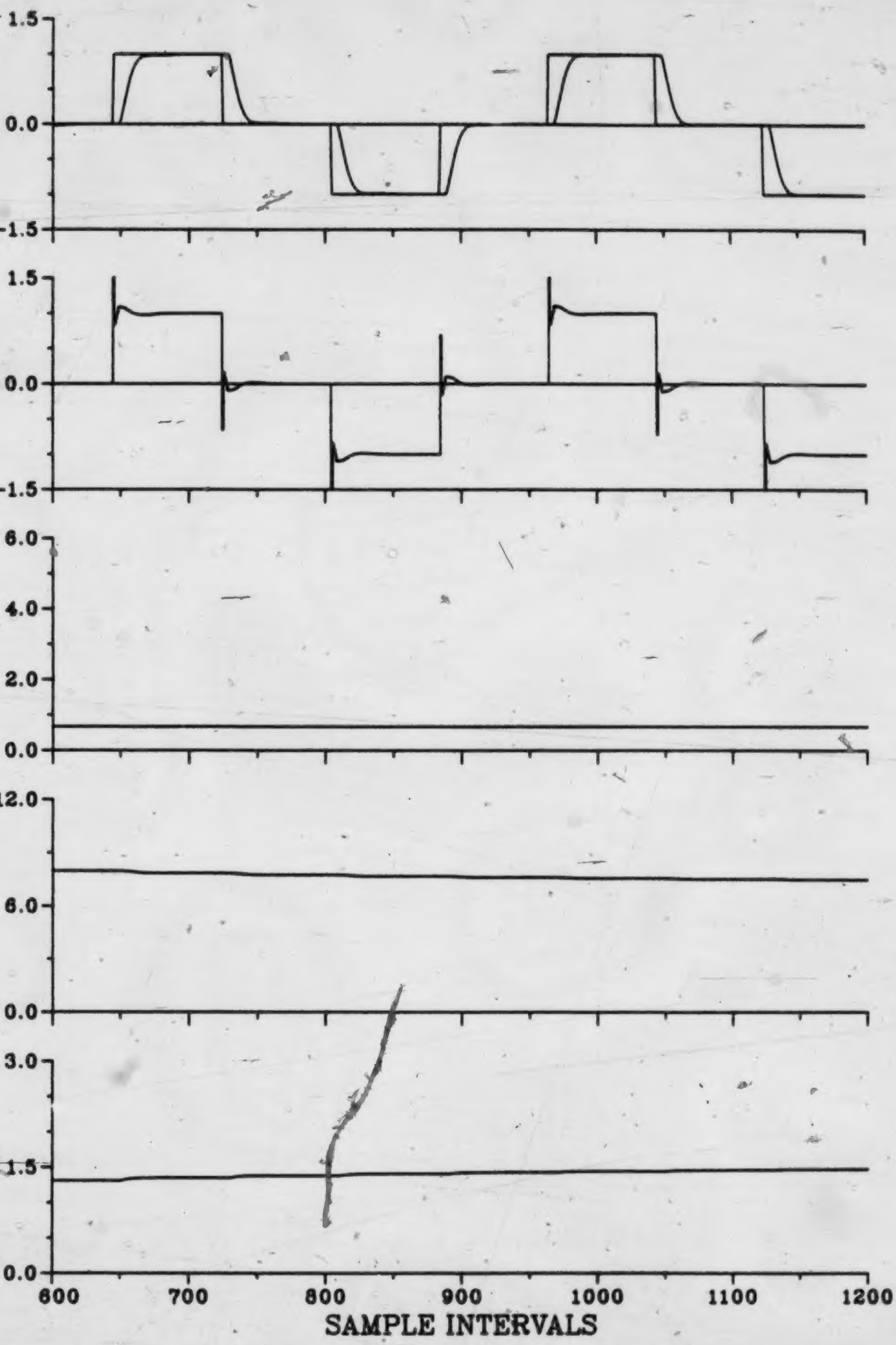

Figure 8.4 Servo self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=40,da=de=4)

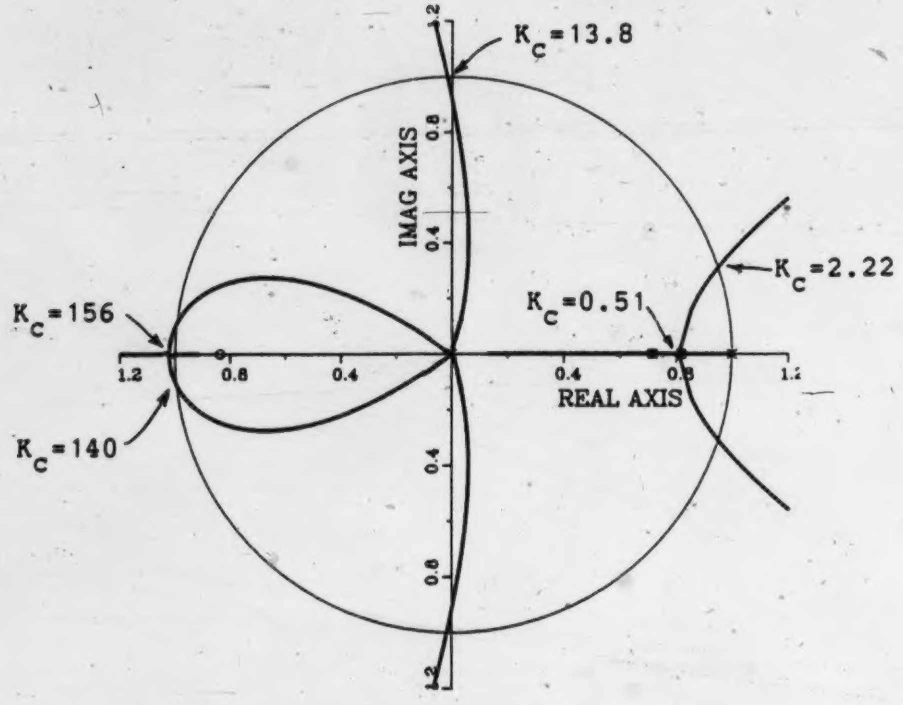

Figure 8.5 Root loci of Process Model 2 with selt-tuning PID controller for time delay = 4

8.5. The K_c^u is reduced further and the breakaway point is moved toward (1,j0).

The overall performance of the self-tuning controller is excellent if the time delay is known. The controller gain calculation method takes the time delay into account when determining the controller gain. The results for a mismatch in time delay estimate are poor during the tuning phase. Incremental variable identification with this mismatch is difficult. Background initial process parameter tuning is recommended before self-tuning control is implemented.

A similiar series of simulation runs are presented for Process Model 4 where a mismatch between process and controller order exists. In Figure 8.6 the closed-loop response for a known time delay of d=1 is shown. The response is slow and overdamped but reasonable. A faster response may be achieved by increasing the M_p setpoint. Figure 8.7 shows the effect of time delay mismatch for PM4. Underestimation of the time delay in addition to the process/model order mismatch results in control law parameters which are unreasonable (i.e. negative T_i and T_d). The estimation of a reasonable model of the process by incremental variable identification has failed. Figure 8.8 considers a time delay mismatch where the actual delay is d=0 but is estimated to be between d=0 and d=1. The controller produces a reasonable underdamped response. Three b_i parameters are estimated and all are

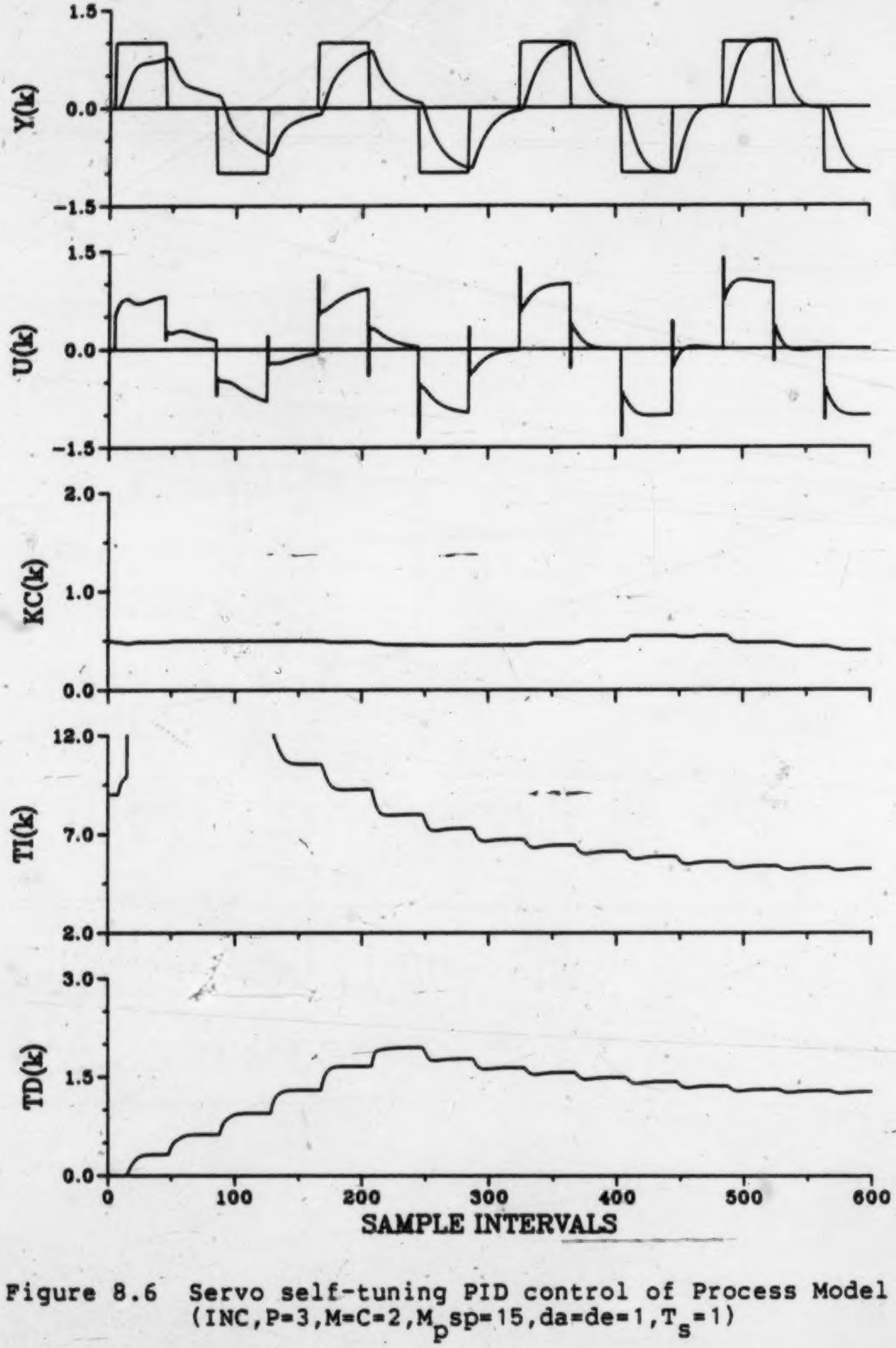

Figure 8.6 Servo self-tuning PID control of Process Model 4 (INC, $P=3, M=C=2, M_p sp=15, da=de=1, T_s=1$)

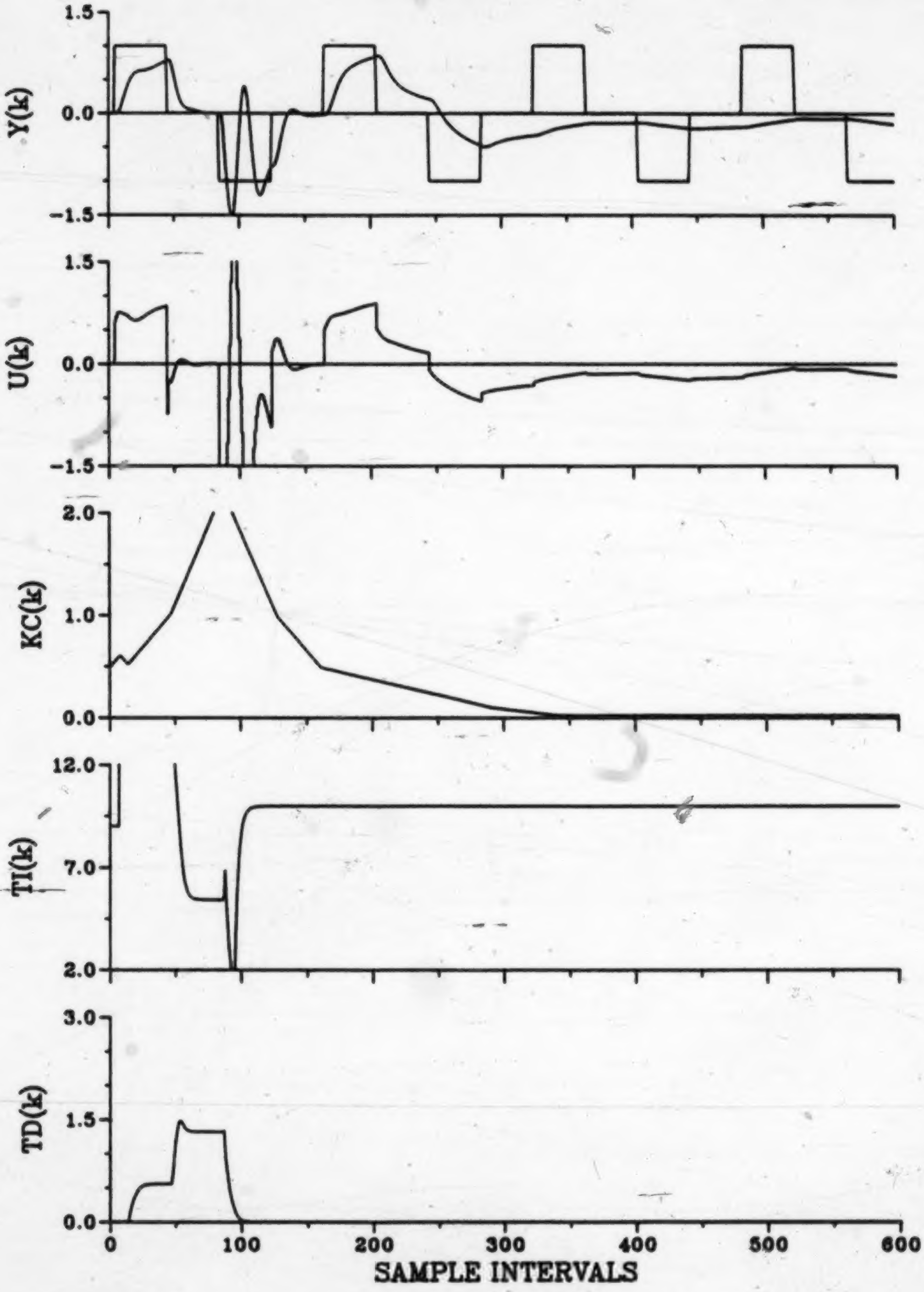

Figure 8.7 Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=15,da=1,de=0,T_s=1)

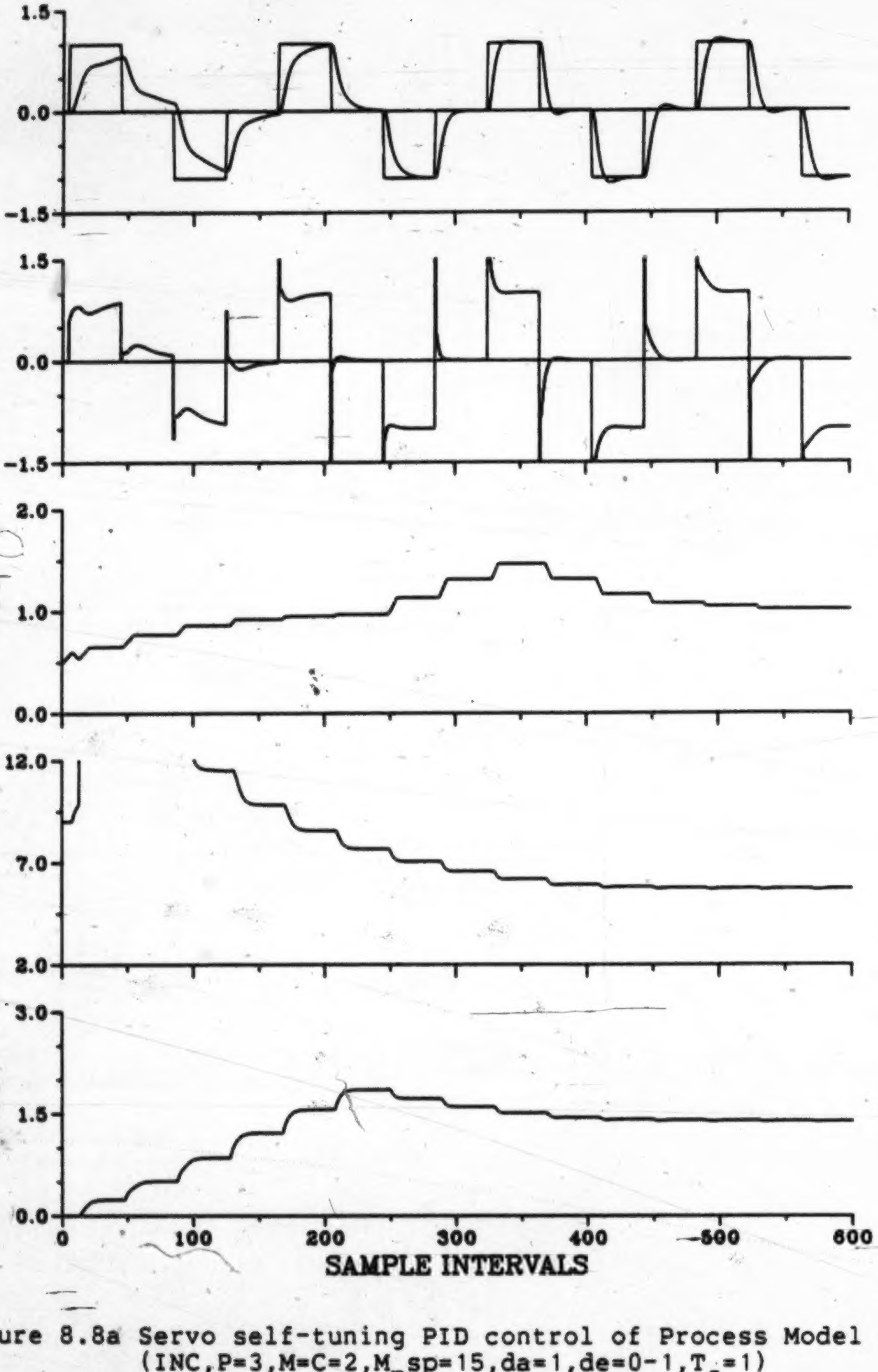

Figure 8.8a Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=15,da=1,de=0-1,T_s=1)

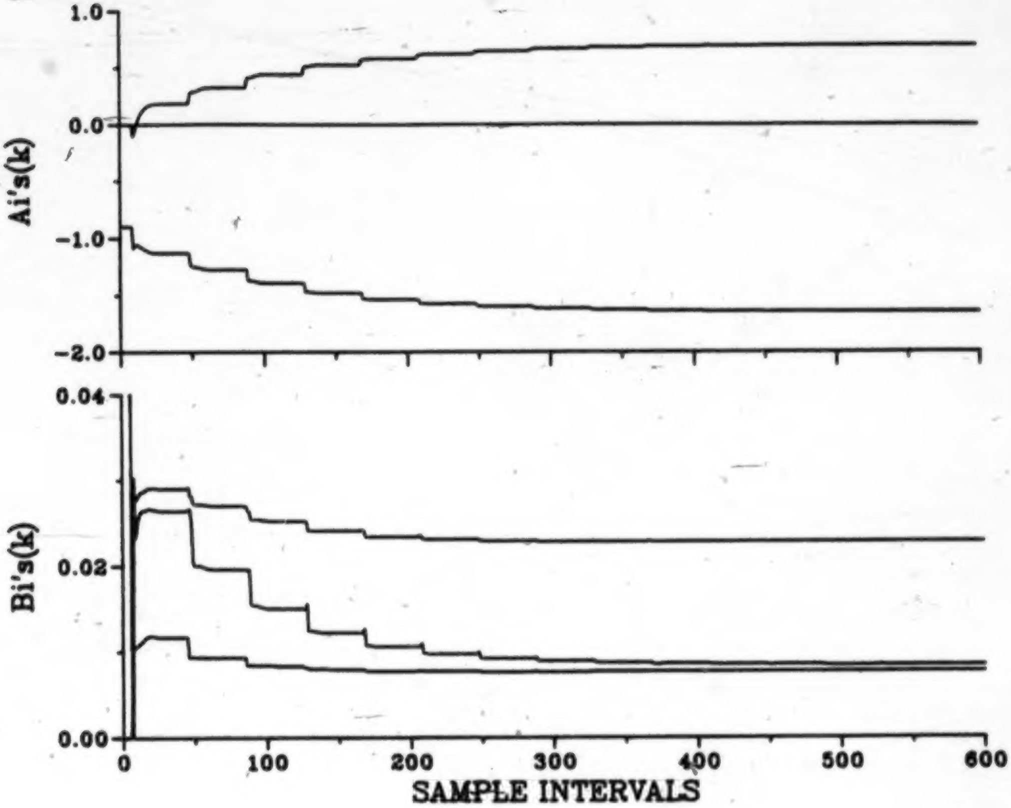

Figure 8.8b Servo self-tuning PID control of Process Model 4 (INC, $P=3, M=C=2, M_p sp=15, da=1, de=0-1, T_s=1$)

nonzero since the process is third order.

The performance of the self-tuning PID controller for a known time delay of d=4 sample periods is shown in Figure 8.9. The simulation run length was 1200 sample intervals but only the last 600 are shown. The closed-loop response is slow and underdamped with a large overshoot which could be reduced by lowering the M_p setpoint. Recall that PM4 is sampled at rate such that the high frequency component of the continuous model is significant. If PM4 is sampled at T_s=2, this high frequency mismatch is reduced. The effect of this change for a d=4 is shown in Figure 8.10. The closed-loop response is excellent. Note that the actual delay is twice that of Figure 8.9. This good performance is attributed to a better estimated model by incremental variable identification.

Overall the self-tuning PPPID controller gives reasonable control for a process/model order mismatch and known time delays. If the delay is unknown, it is better to overestimate the delay and estimate extra b_i parameters. However, excessive overestimation leads to poorer conditioning of the estimation algorithm.

8.2.2 Time Delay Compensation

Time delay compensation (TDC) is a second control technique for handling processes dominated by time delays. The technique generally involves prediction of the undelayed output of the process. This predicted output is

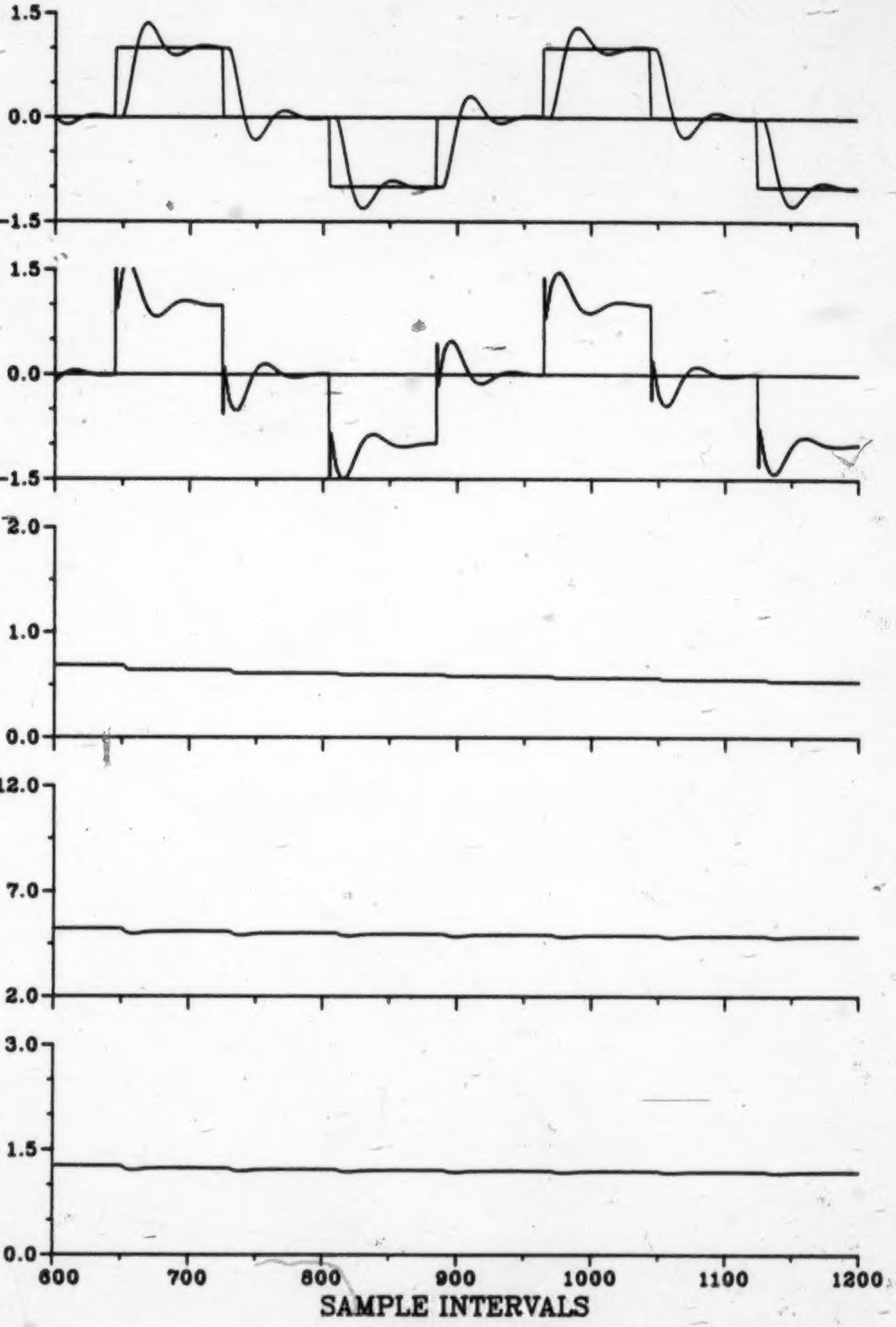

Figure 8.9 Servo self-tuning PID control of Process Model 4 (INC, P=3, M=C=2, $M_p$sp=40, da=de=4, T_s=1)

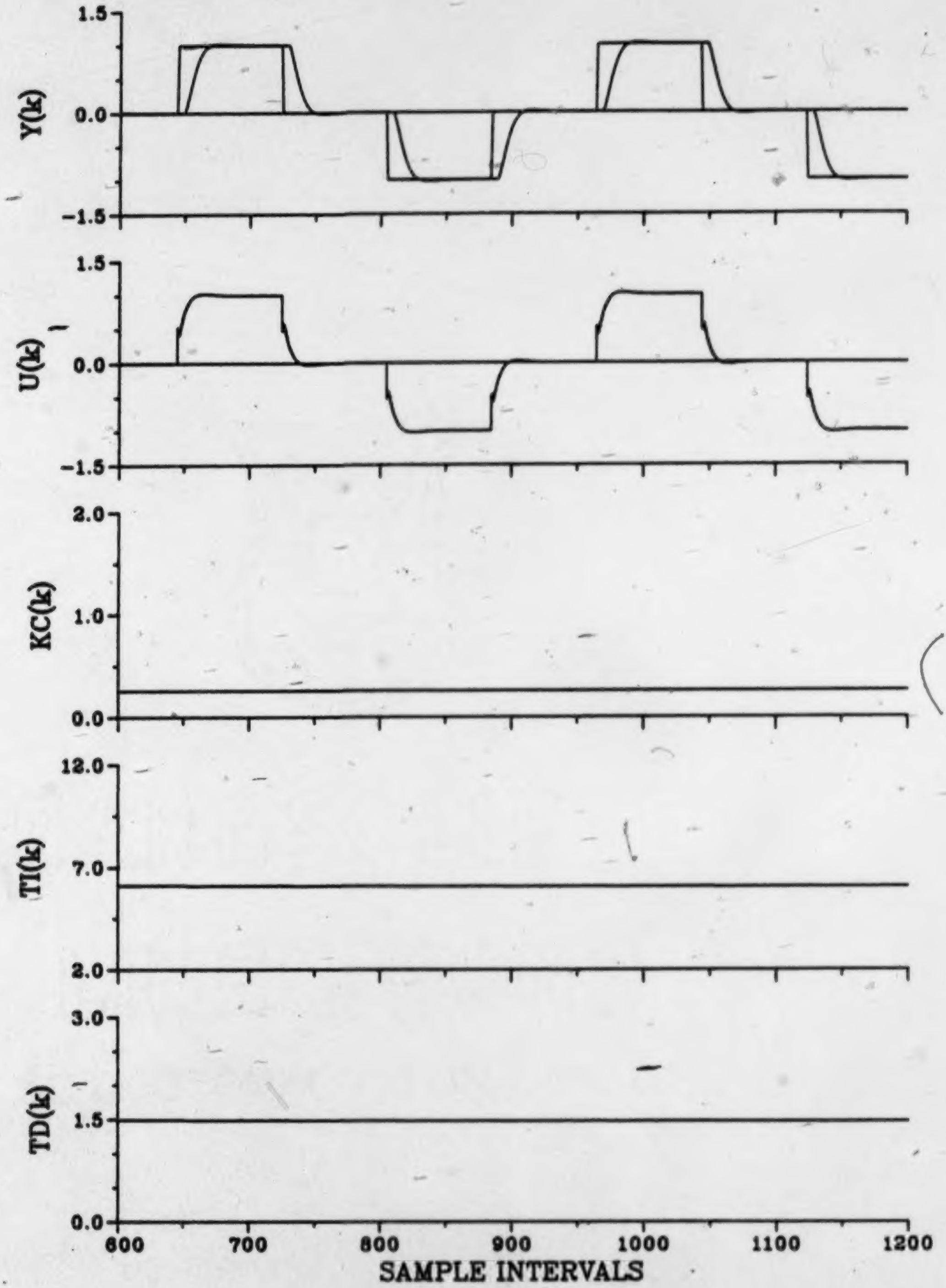

Figure 8.10 Servo self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=40,da=de=4,T_s=2)

then fedback to the controller for control output calculation. The particular method used here is the model-based, adaptive time delay compensation (TDC) of Vogel and Edgar (1980). This adaptive TDC is used in conjunction with the self-tuning PPPID control law for a no process/model order mismatch case (PM2) and a process/model order mismatch case (PM4).

The TDC simulation studies presented here are based on a strategy where the adaptive TDC and the self-tuning controller are considered to be totally separate. The TDC estimates a process model and estimates the undelayed process output. The self-tuning takes this predicted output and estimates a second model for the controller design calculations. Therefore, two process models are estimated. The advantage of separating the TDC design from the self-tuning PID design is that both are independent of the other. For example a nonadaptive or non-model-based time delay compensator can be used with the self-tuning PID controller. A second approach is to use an integrated time delay compensator and self-tuning PID controller where only one process model is estimated and this one model used to design the TDC and the self-tuning controller. This would reduce computational effort and speed convergence of the controller parameters. Clearly, the first approach gives more flexibility but at the expense of being more difficult.

The simulation results are summarized in Table 8.2 for the two processes considered. Figure 8.11 shows the results

Table 8.2 Performance of self-tuning PID control in the presence of time delay compensation (TDC)

Process	Mismatch		Figure	$\frac{M_p}{SP}$	Time-Domain Response			Comments
Model	Order	Delay	No.		M_p	t_r	t_s	
PM2	P=2 M=2 C=2	da=4 de=4	8.11	15%	10%	7.9	17.7	Excellent underdamped response
		da=2,4 de=2-5	8.12	15%	42%	8.6	45^+	Highly underdamped response
PM4 (T_s=1)	P=3 M=2 C=2	da=4 de=4	8.13	30%	50%	-	-	Poor underdamped response
PM4 (T_s=2)	P=3 M=2 C=2	da=4 de=4	8.14	30%	13%	16.3	68.0	Reasonable underdamped response
		da=2,4 de=2-5	8.15	30%	30%	17.6	72.4	Highly underdamped response

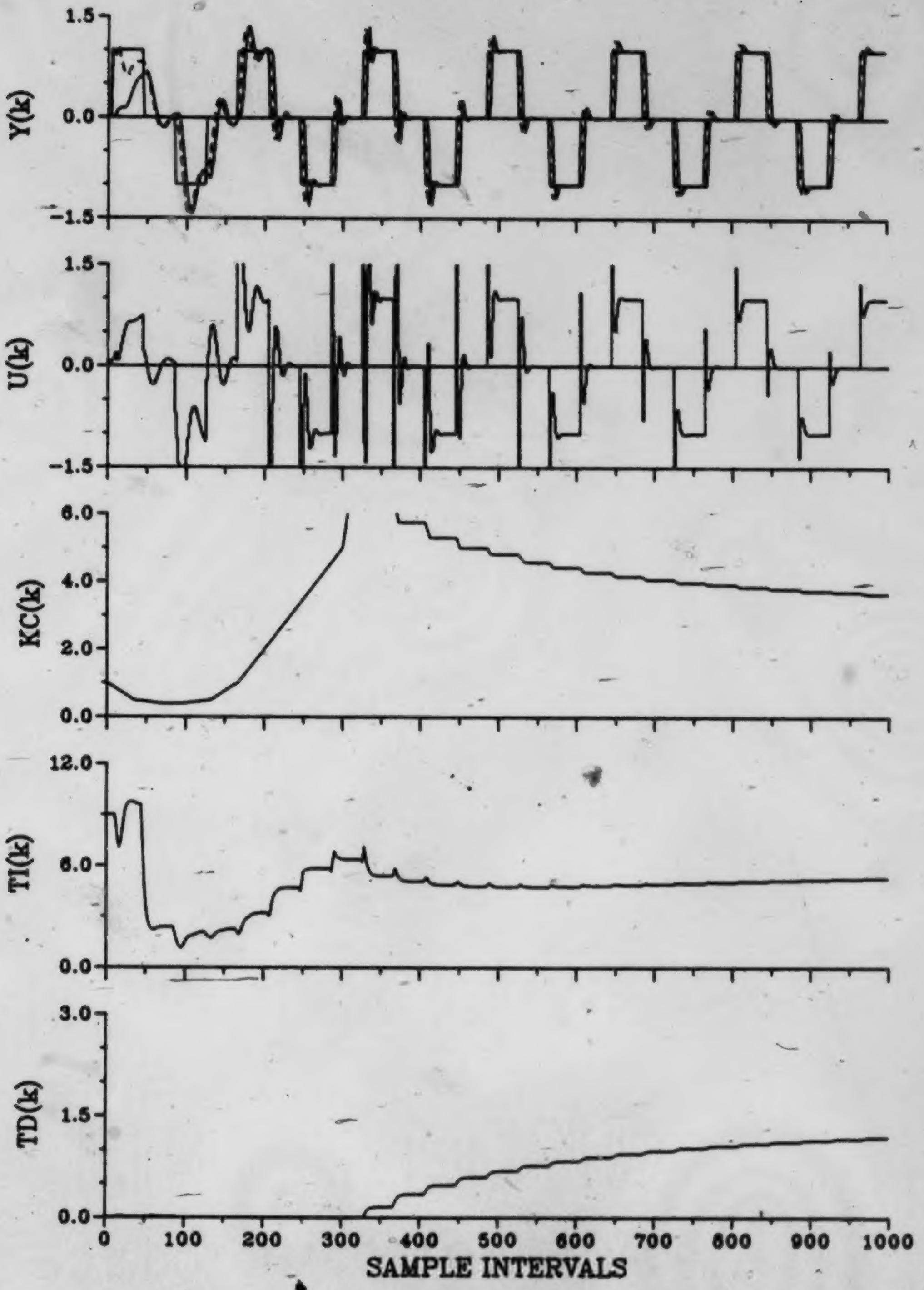

Figure 8.11a Servo self-tuning PID control of Process Model 2 with time delay compensator (---- Ytdc(k)) (INC,P=M=C=2,$M_p$sp=15,da=de=4,TDC)

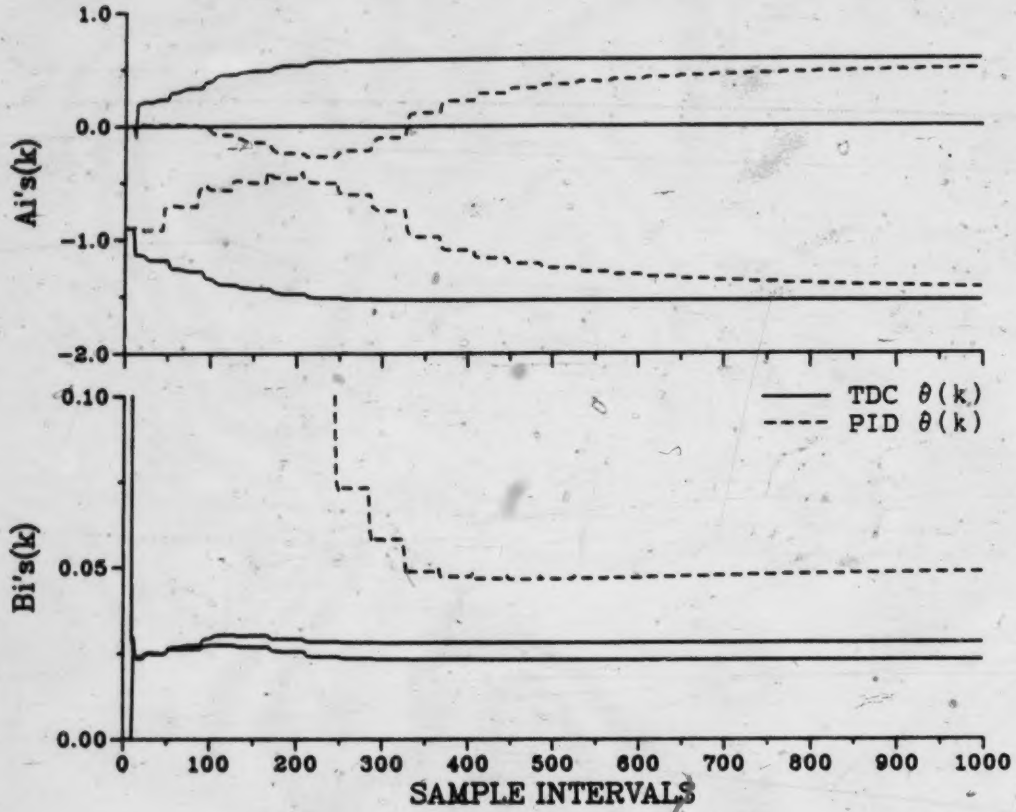

Figure 8.11b Servo self-tuning PID control of Process Model 2 with time delay compensator (INC,P=M=C=2,$M_p$sp=15,da=de=4,TDC)

for Process Model 2 with a known time delay of d=4 sample periods. Closed-loop response is excellent. The undelayed predicted output, marked by the dashed line, leads the actual output. The convergence of the estimated process parameters for controller design are slow to converge as they are dependent on the TDC process parameter convergence.

Figure 8.12 shows the results of the self-tuning PID controller with TDC for an unknown varying time delay. The run is initiated with tuned parameters for an actual delay of d=2 and an estimated delay ranging from d=2-5. After 250 sample intervals the time delay is changed to d=4. The closed-loop response is highly underdamped but the overshoot is being reduced as the TDC process parameter estimates adapt to the new time delay.

The performance of the self-tuning PID controller with the TDC is shown in Figure 8.13 for Process Model 4 with a known time delay of 4 sample periods. The closed-loop response for this process/model order mismatch case is poor. The response is characterized by large overshoots and long settling times. The cumulative effect of time delays, sequential estimation of two process models and the high frequency process/model mismatch results in poor closed-loop performance. To reduce the mismatch of high frequency dynamics, PM4 is sampled at $T_s=2$ instead of $T_s=1$. The time delay is again chosen to be 4 sample periods (i.e. delay=4x2=8 secs). The improvement in performance is dramatic as is seen in Figure 8.14. The improvement is

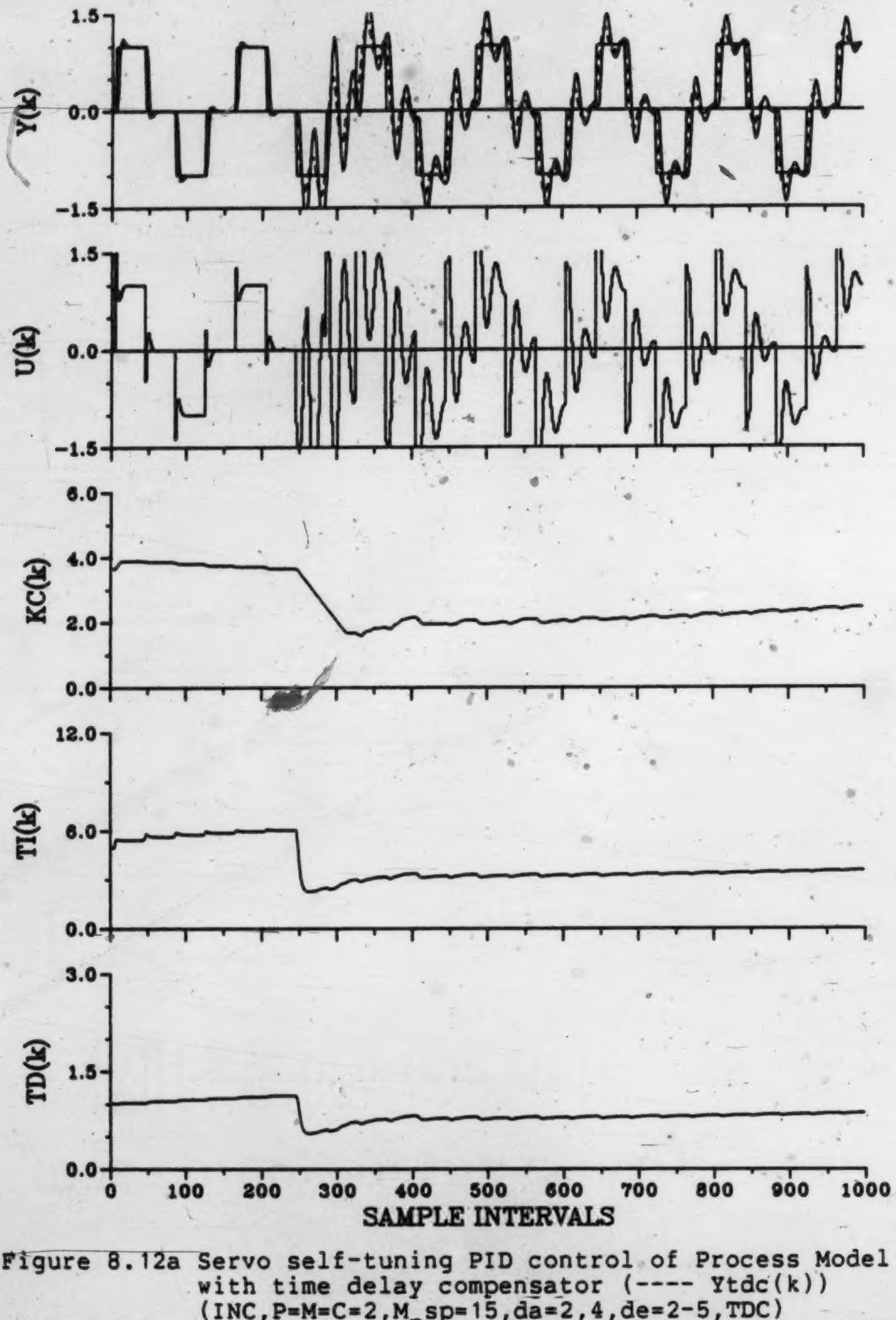

Figure 8.12a Servo self-tuning PID control of Process Model 2 with time delay compensator (---- Ytdc(k)) (INC,P=M=C=2,$M_p$sp=15,da=2,4,de=2-5,TDC)

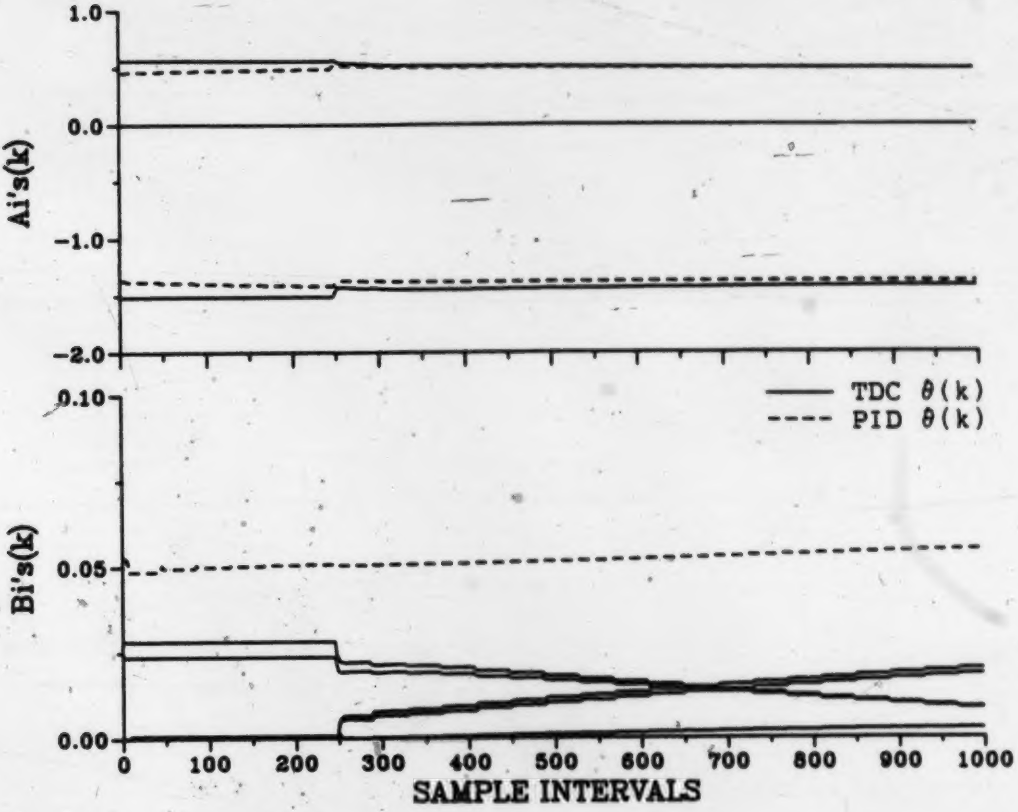

Figure 8.12b Servo self-tuning PID control of Process Model 2 with time delay compensator (INC,P=M=C=2,$M_p$sp=15,da=2,4,de=2-5,TDC)

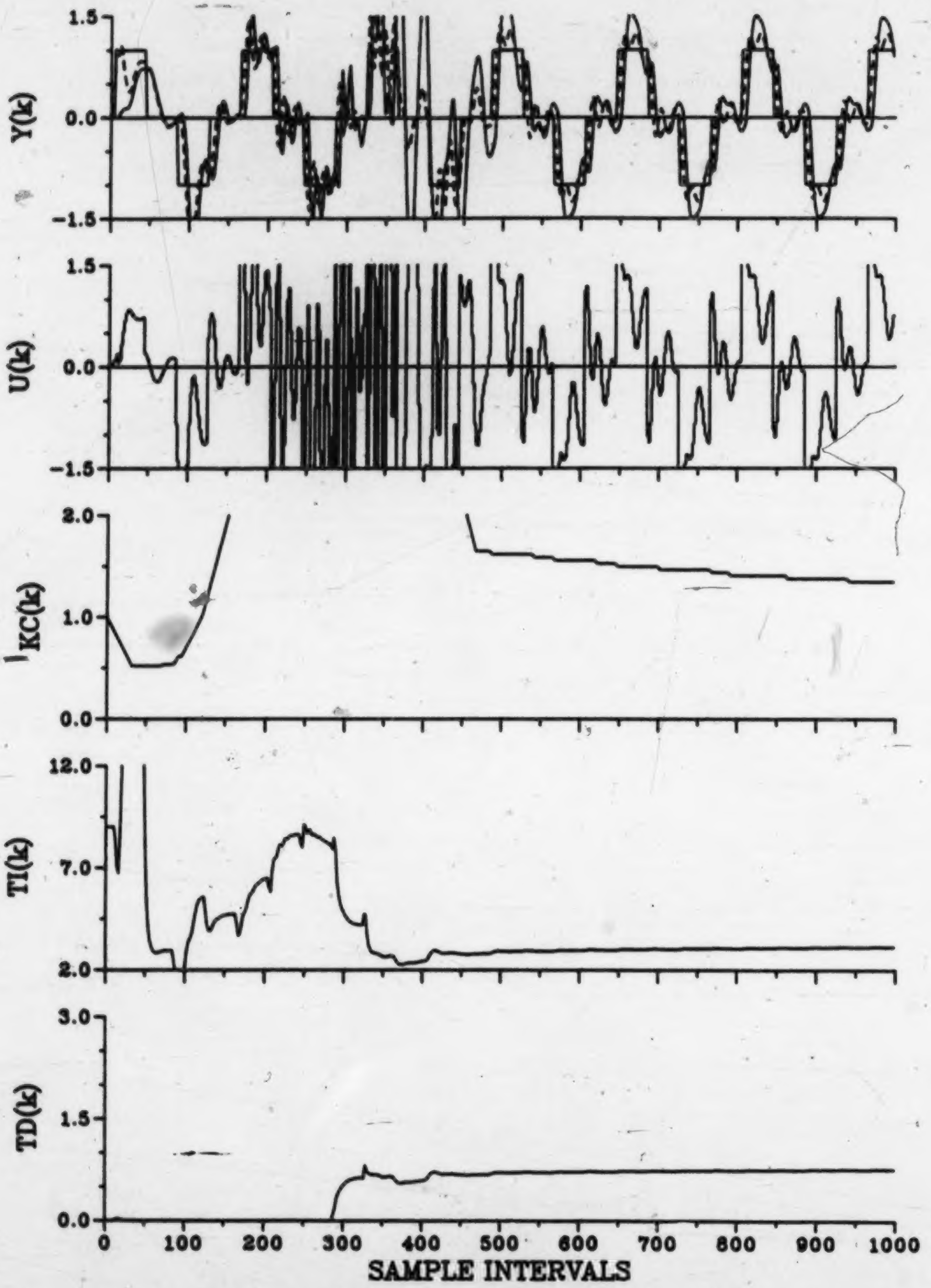

Figure 8.13a Servo self-tuning PID control of Process Model 4 with time delay compensator (---- Ytdc(k)) (INC,P=3,M=C=2,$M_p$sp=30,da=de=4,TDC,T_s=1)

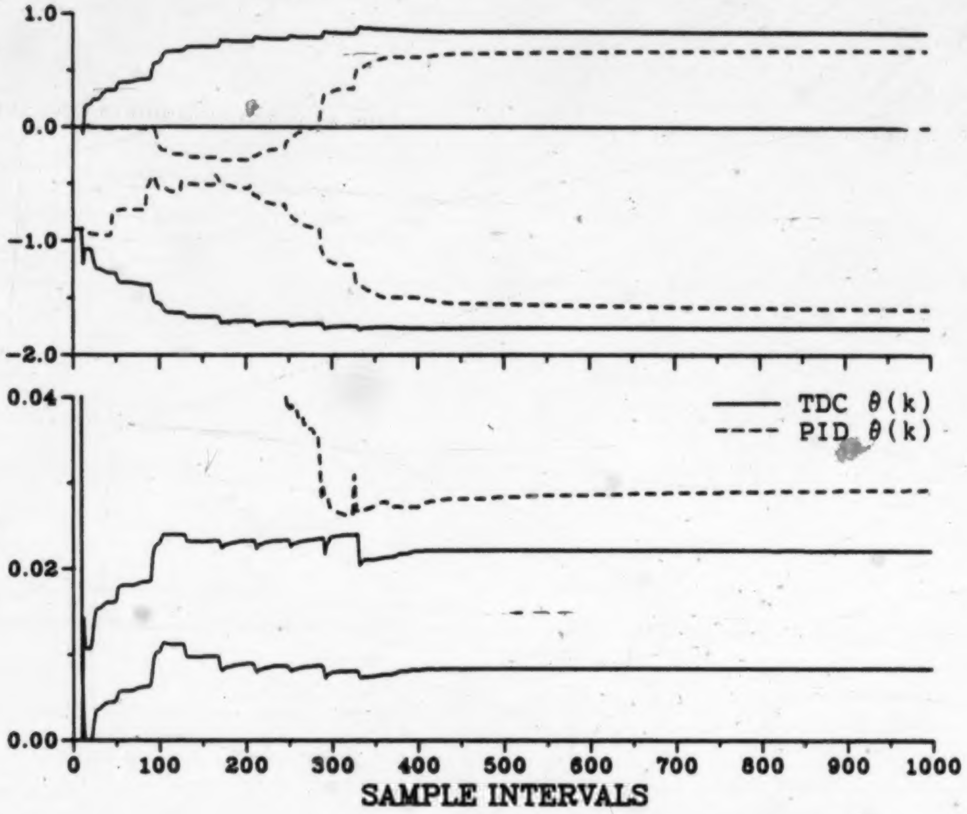

Figure 8.13b Servo self-tuning PID control of Process Model 4 with time delay compensator (INC,P=3,M=C=2,$M_p$sp=30,da=de=4,TDC,T_s=1)

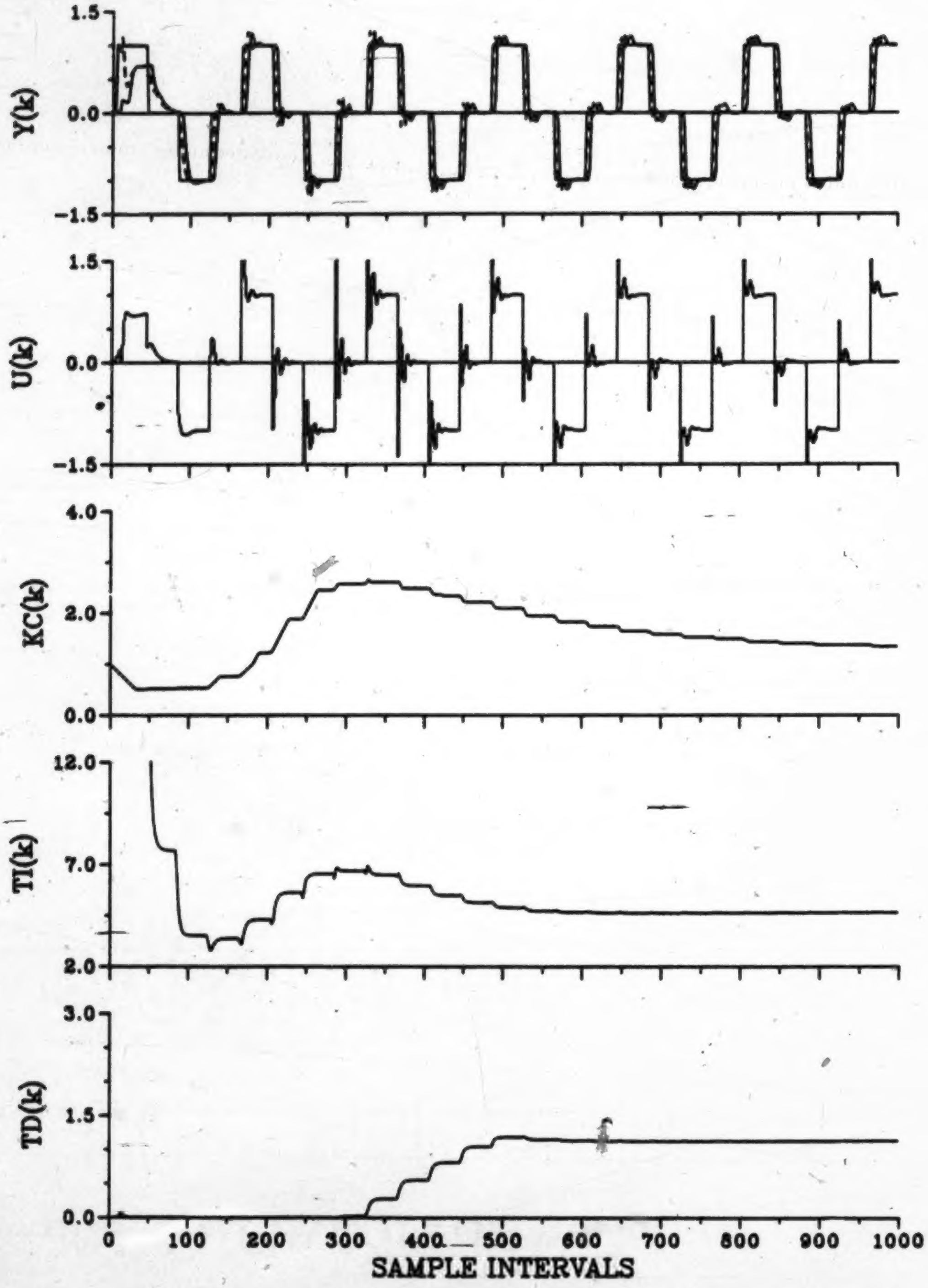

Figure 8.14a Servo self-tuning PID control of Process Model 4 with time delay compensator (---- Ytdc(k)) (INC,P=3,M=C=2,$M_p$sp=30,da=de=4,TDC,T_s=2)

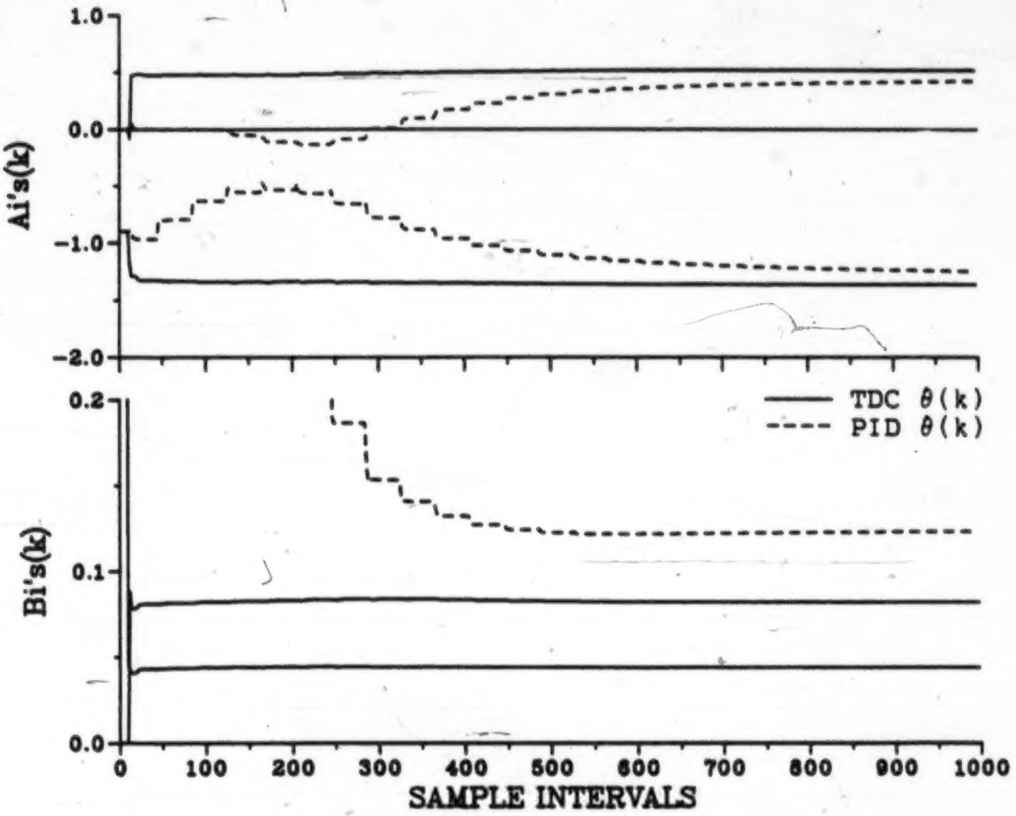

Figure 8.14b Servo self-tuning PID control of Process Model 4 with time delay compensator
($INC, P=3, M=C=2, M_p sp=30, da=de=4, TDC, T_s=2$)

primarily attributed to the better estimated process model. Incremental variable identification is sensitive to the high frequency dynamics present in PM4 when sampled at $T_s=1$. At $T_s=2$ the high frequency dynamics are attenuated and a better estimated process model results.

The performance of the self-tuning PID controller with an unknown varying time delay is shown in Figure 8.15. The run is initialized with tuned parameters for an actual time delay of d=2 and an estimated time delay of d=2-5. The time delay is changed to d=4 after 250 sample intervals. The response is similar to Figure 8.12 for PM2. The overshoot is being reduced as the TDC b_i estimates adapt to the new time delay.

In summary the performance of the self-tuning PID controller with the Vogel and Edgar time delay compensator is good. The convergence of the estimated parameters could be improved if an integrated strategy were adopted. It is again shown that the sampling rate choice is an important parameter for the performance of incremental variable identification in the presence of high frequency dynamic mismatch.

8.3 Simulation Results for Deterministic Disturbances

Adaptive control on real processes generally requires process model estimation in the presence of deterministic disturbances. Two disturbances most often encountered are process bias caused by nonzero mean operating levels of

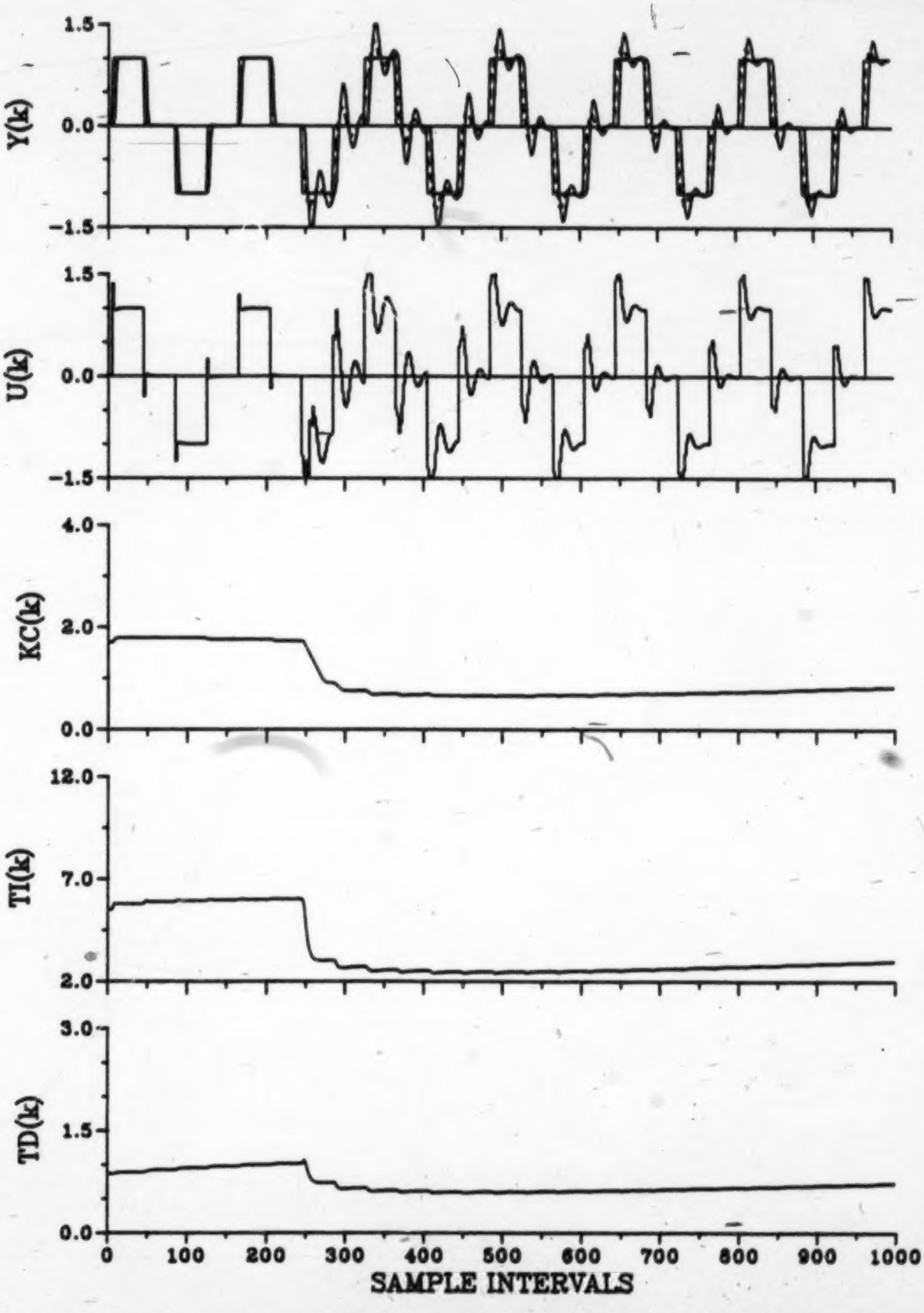

Figure 8.15a Servo self-tuning PID control of Process Model 4 with time delay compensator (---- Ytdc(k)) (INC,P=3,M=C=2,$M_p$sp=30,da=2,4,de=2-5,TDC,T_s=2)

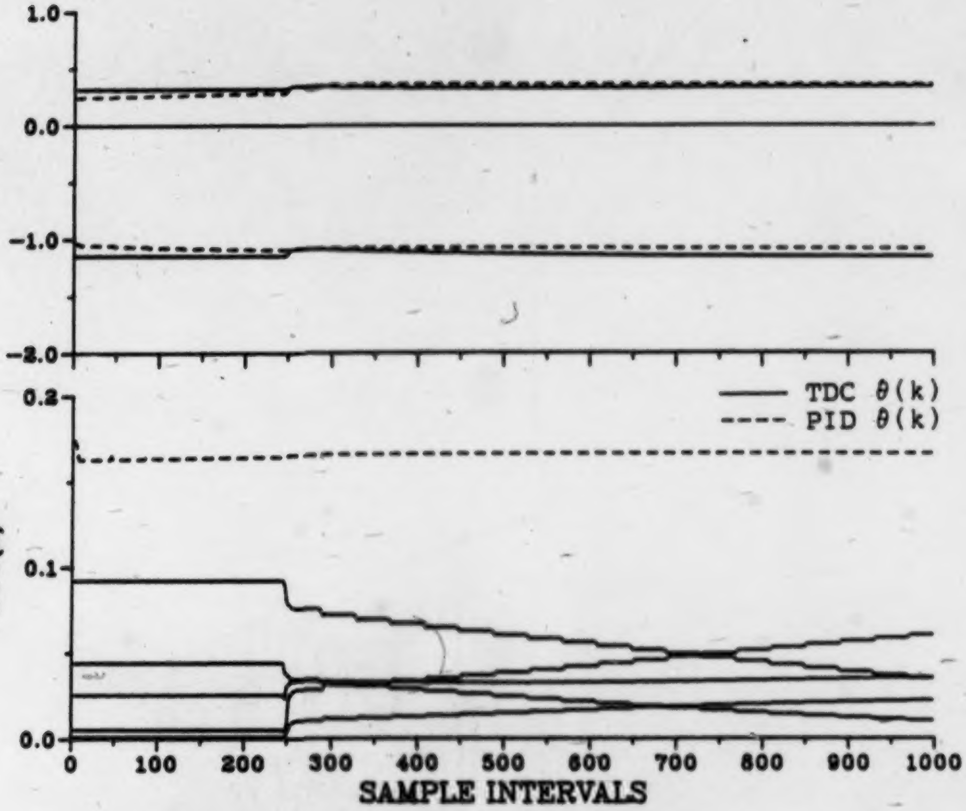

Figure 8.15b Servo self-tuning PID control of Process Model 4 with time delay compensator (INC,P=3,M=C=2,$M_p$sp=30,da=2,4,de=2-5,TDC,T_s=2)

plant signals and load disturbances. The following results examine the performance of the self-tuning PI(D) controller based on incremental variable identification under such conditions. These results are compared with some selected results based on positional variable identification.

8.3.1 Process Bias

A process bias generally occurs when the mean level of the process input and output signals are nonzero. The actual value of this bias is calculated by Equation 4.20. From this equation it is evident that if the input and output levels are zero mean, the process bias is zero. The effect of this bias on the performance of the self-tuning PI controller for incremental and positional parameter estimates is summarized in Table 8.3. Four runs are presented for both Process Model 2 and Process Model 4. The first two runs are based on zero mean data; the second 2 runs on nonzero mean data. The nonzero mean levels are chosen to be the mean top composition in weight percent (y_{ss}=95.0) and the reflux flow rate in grams per second (u_{ss}=9.93) for a distillation column simulation model used in Chapter 9. These numbers provided proper scaling of the data.

Both reflux and overhead composition move between ±1.0 of the mean value and thus do not create numerical problems for the estimation algorithm. The resultant bias for PM2 and PM4 is 4.36 and 2.76 respectively. The performance

Table 8.3 Performance of self-tuning PI control for processes with nonzero mean levels

Process Model	Mismatch	Figure No.	M_p SP	Time-Domain Response			ID Type	Process Bias	Comments
				M_p	t_r	t_s			
PM2	P=2 M=2 C=1	7.6	15%	6%	7.5	27.7	INC	0.0	Reasonable underdamped response
		8.16	15%	5%	7.8	28.3	POS	0.0	Similiar to Figure 7.6, slightly faster parameter convergence
		8.17	15%	6%	7.5	27.7	INC	4.36	Response identical to Figure 7.6
		8.18	15%	0%	26.4	40^+	POS	4.36	Poor overdamped response, biased parameter estimates, slow bias estimate convergence
PM4	P=3 M=2 C=1	7.8	15%	0%	30.3	46.0	INC	0.0	Slow underdamped response
		8.19	15%	0%	15.9	45.0	POS	0.0	Slow underdamped response
		8.20	15%	0%	$45.^+$	-	POS	2.76	Poor overdamped response, biased parameter estimates, slow bias estimate convergence

criteria for Table 8.3 are calculated after the self-tuning controller has tuned for 400 sample intervals.

Figure 7.6 provides the PM2 base case results based on incremental variable identification and no process bias. The equivalent result for positional variable identification is shown in Figure 8.16. The closed-loop response is very similar to Figure 7.6 with a slightly faster parameter convergence observed. The similarity of performance results from both identification methods converging to the true parameter values in the absence of mismatch and process bias. The effects of a nonzero process bias are shown in Figures 8.17 and 8.18. The incremental results seen in Figure 8.17 are identical to the results of Figure 7.6. The presence of process bias does not affect the process parameter estimates based on incremental data as this estimation technique differences the data and $\Delta d=0$. The effect of a process bias on positional estimation is significant, as seen in Figure 8.18. The estimated process parameters are biased as a result of poor convergence of the process bias estimate. After 450 sample intervals the bias estimate is only 3.25. This also results in the poor convergence of the b_i parameters. The closed-loop response is highly overdamped as a result of poor dynamic and static process parameter estimates. Improved performance can be achieved by increasing the M_p setpoint or beginning the run with a better initial estimate of the process bias.

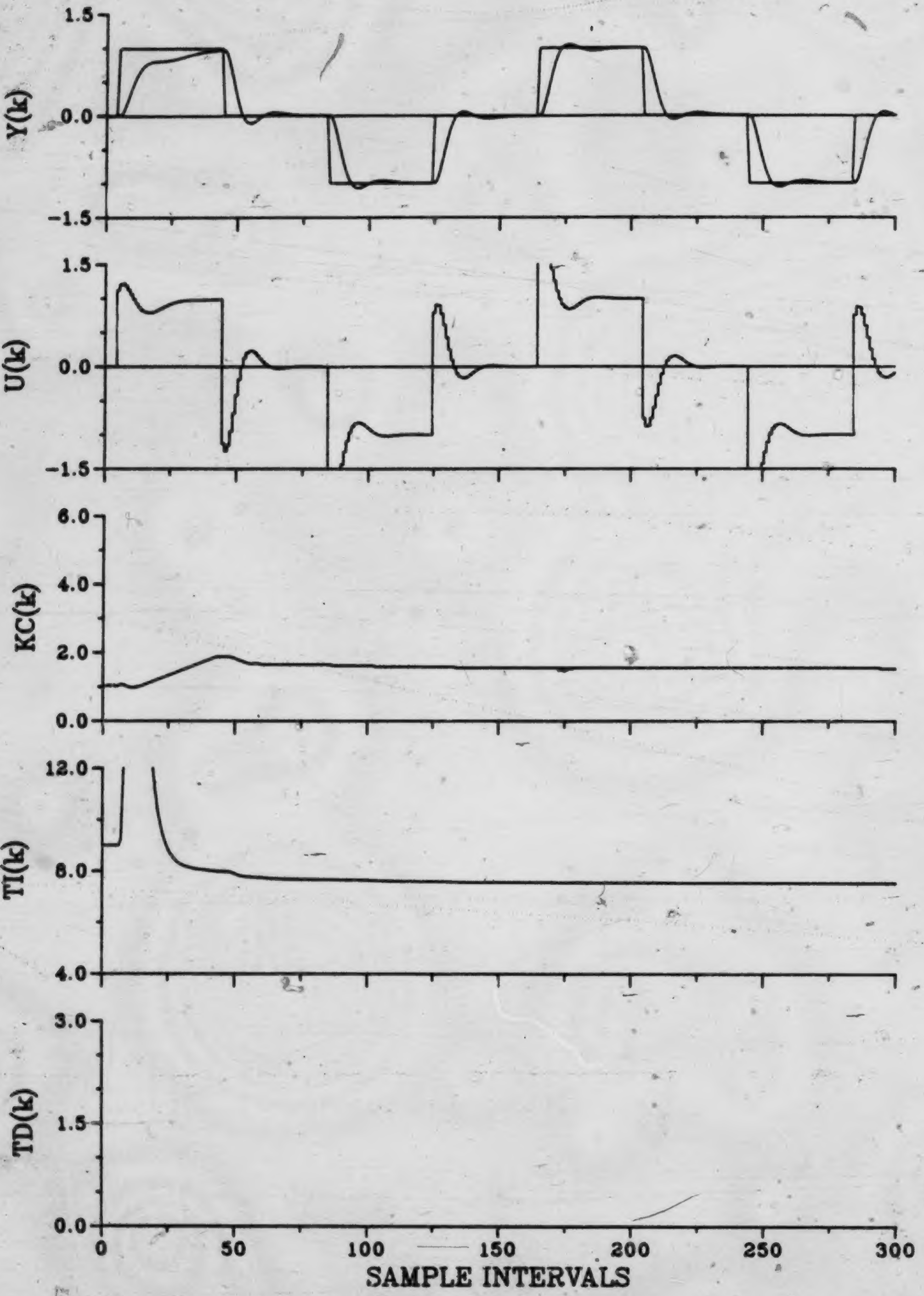

Figure 8.16 Servo self-tuning PI control of Process Model 2 (POS, P=M=2, C=1, $M_p$sp=15, bias=0.0)

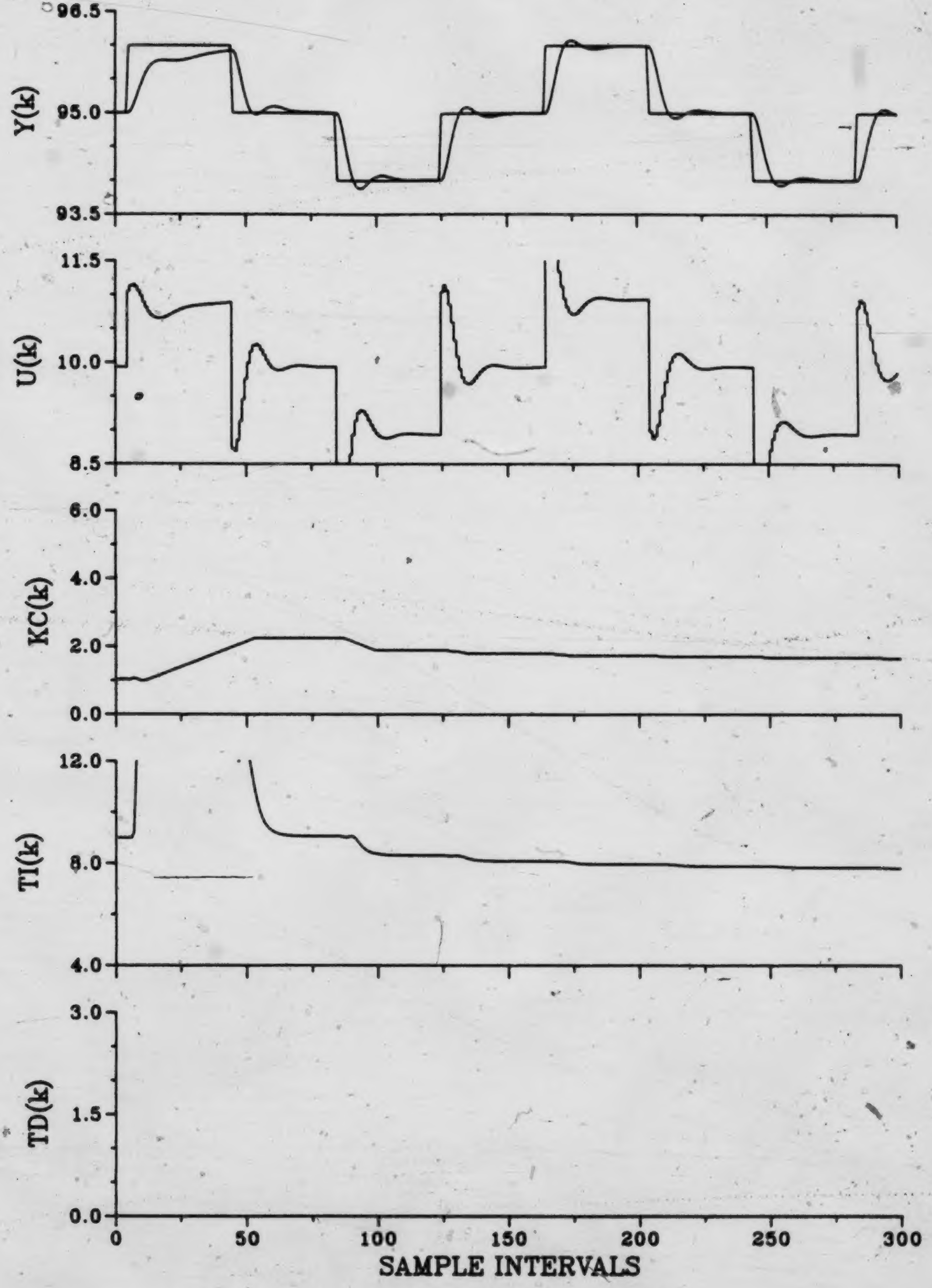

Figure 8.17 Servo self-tuning PI control of Process Model 2 (INC,P=M=2,C=1,$M_p$sp=15,bias=4.36)

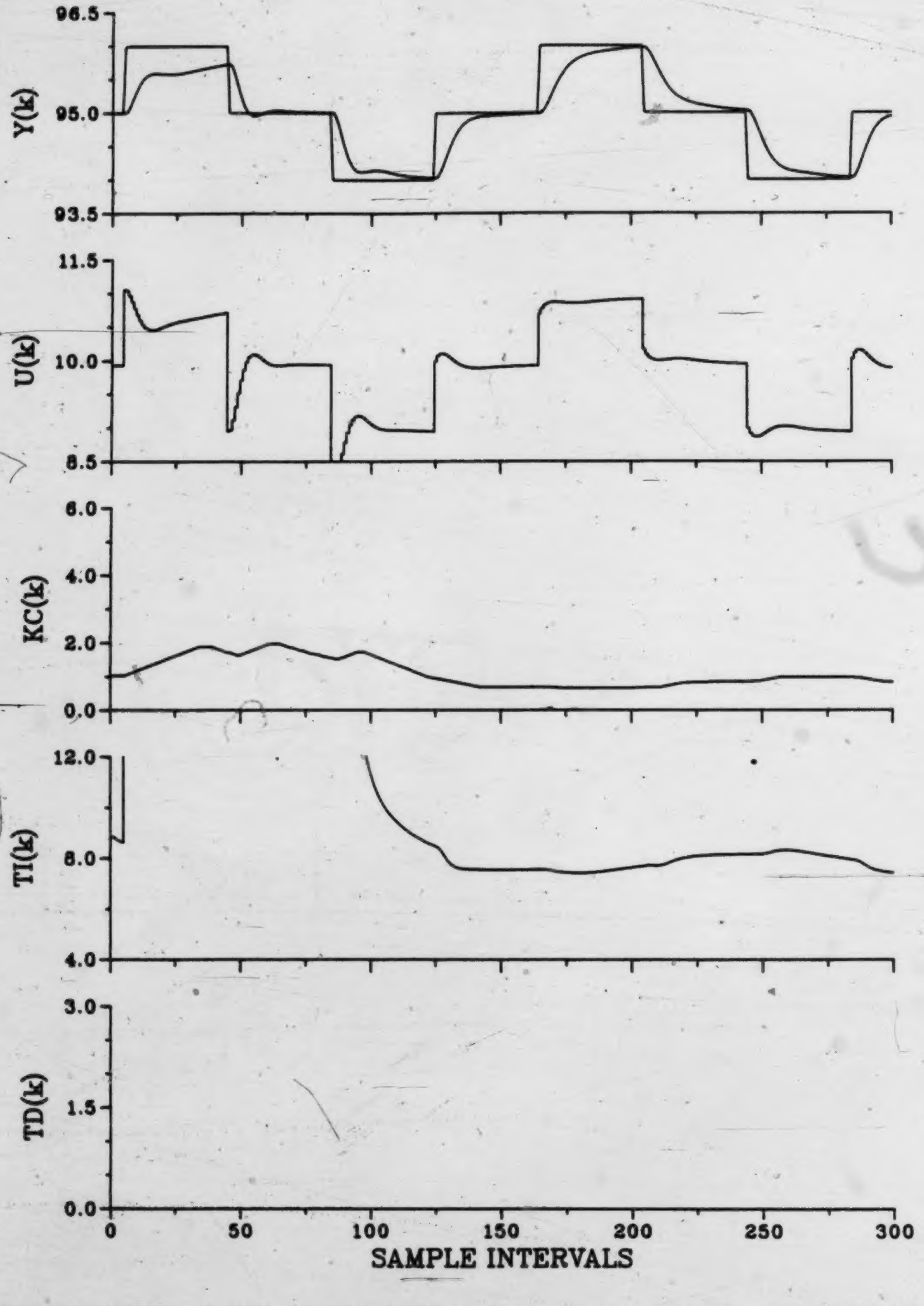

Figure 8.18a Servo self-tuning PI control of Process Model 2 (POS, P=M=2, C=1, $M_p$sp=15, bias=4.36)

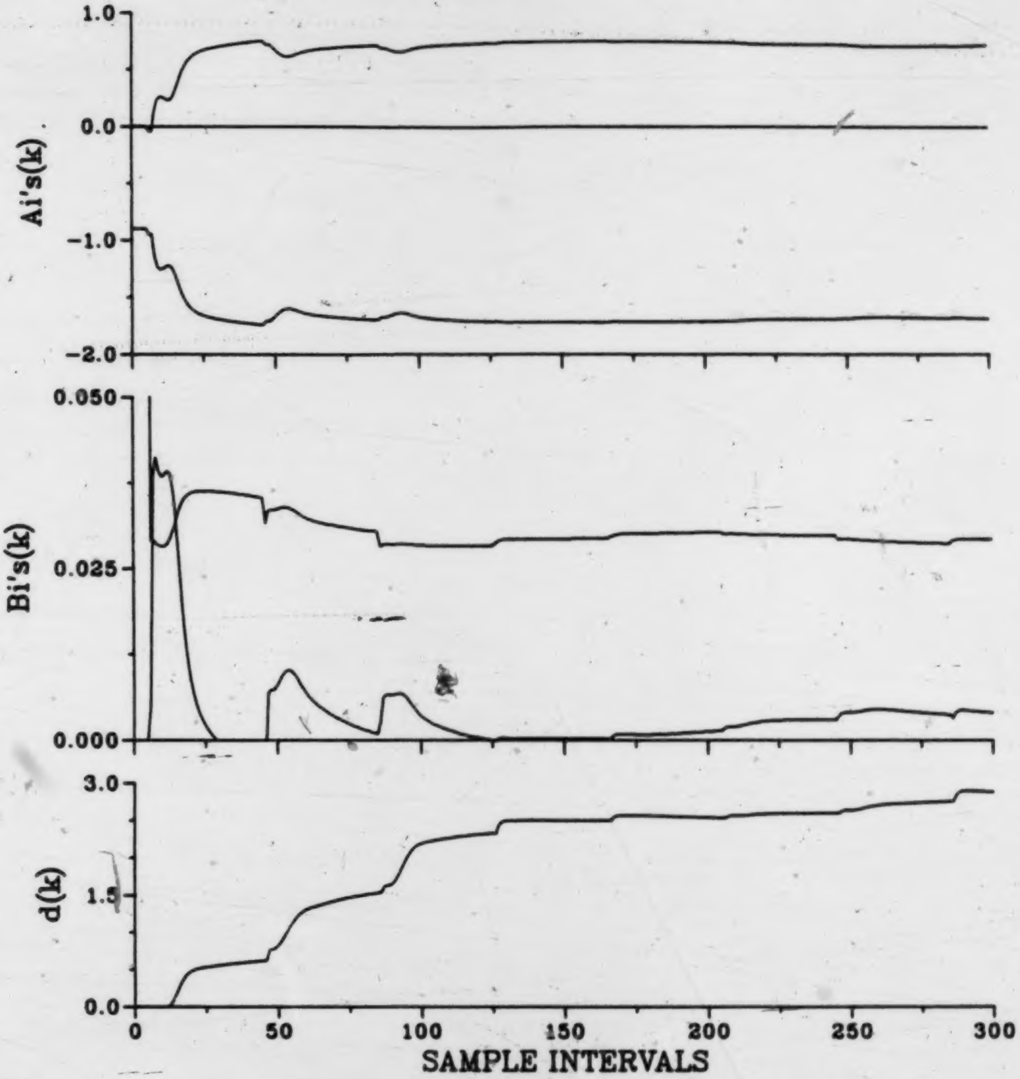

Figure 8.18b Servo self-tuning PI control of Process Model 2 (POS, P=M=2, C=1, $M_p$sp=15, bias=4.36)

The results for PM4 seen in Figures 7.8, 8.19 and 8.20 show the same trends for model estimation with process/model order mismatch. The presence of a process bias has no effect on incremental variable identification therefore the time-domain response is identical to that of the no process bias run. Positional variable identification is characterized by poor convergence of the static and b_i dynamic parameters and biased a_i parameters.

8.3.2 Load Disturbances

Because unmeasurable load disturbances are so common in the process industries, it is necessary to determine their effect on the self-tuning PID controller performance. The objective of this section is to evaluate the regulatory capability of the self-tuning controller tuned for setpoint changes and the effect of unmeasurable load disturbances on closed-loop process parameter estimation for incremental and positional variable identification.

The results presented in this section are for self-tuning PID control of Process Model 2 and Process Model 4. Each simulation run was initialized with a reasonable set of tuned parameters. The load transfer function is chosen as:

$$G(s) = \frac{1}{3s + 1} \tag{8.1}$$

A unit step change in the load variable is made at 50 and

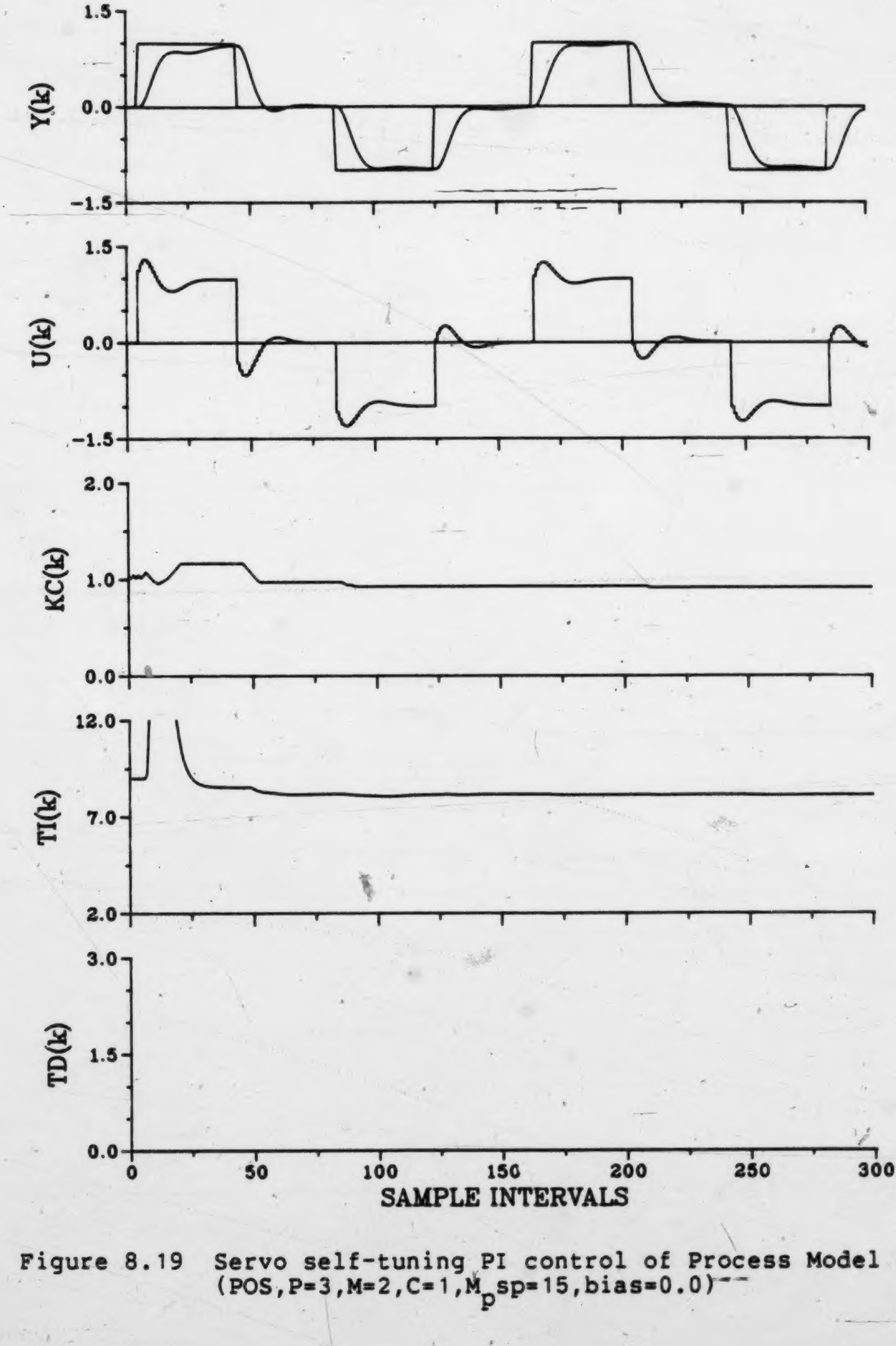

Figure 8.19 Servo self-tuning PI control of Process Model 4 (POS, P=3, M=2, C=1, $M_p$sp=15, bias=0.0)

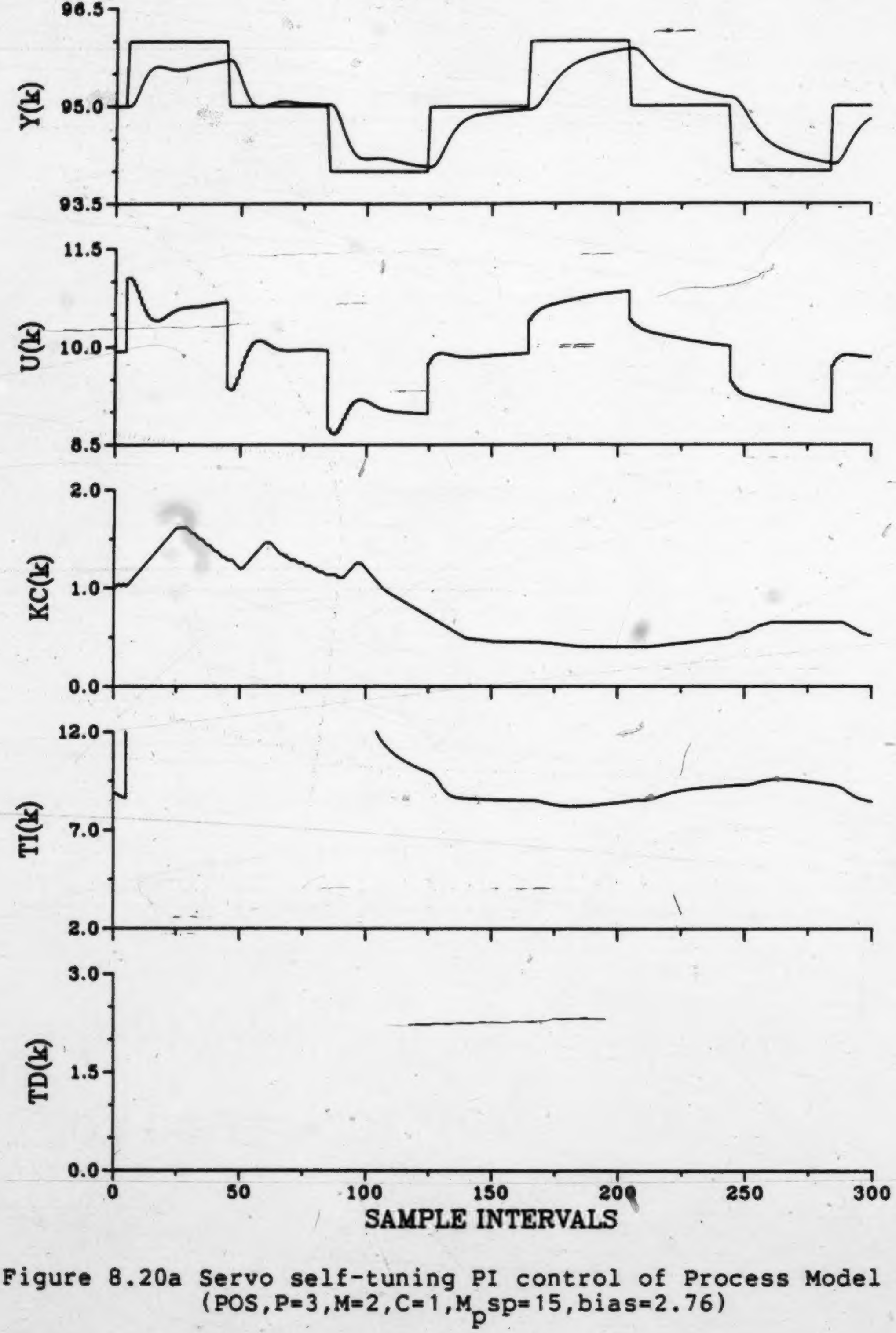

Figure 8.20a Servo self-tuning PI control of Process Model 4 (POS, P=3, M=2, C=1, $M_p$sp=15, bias=2.76)

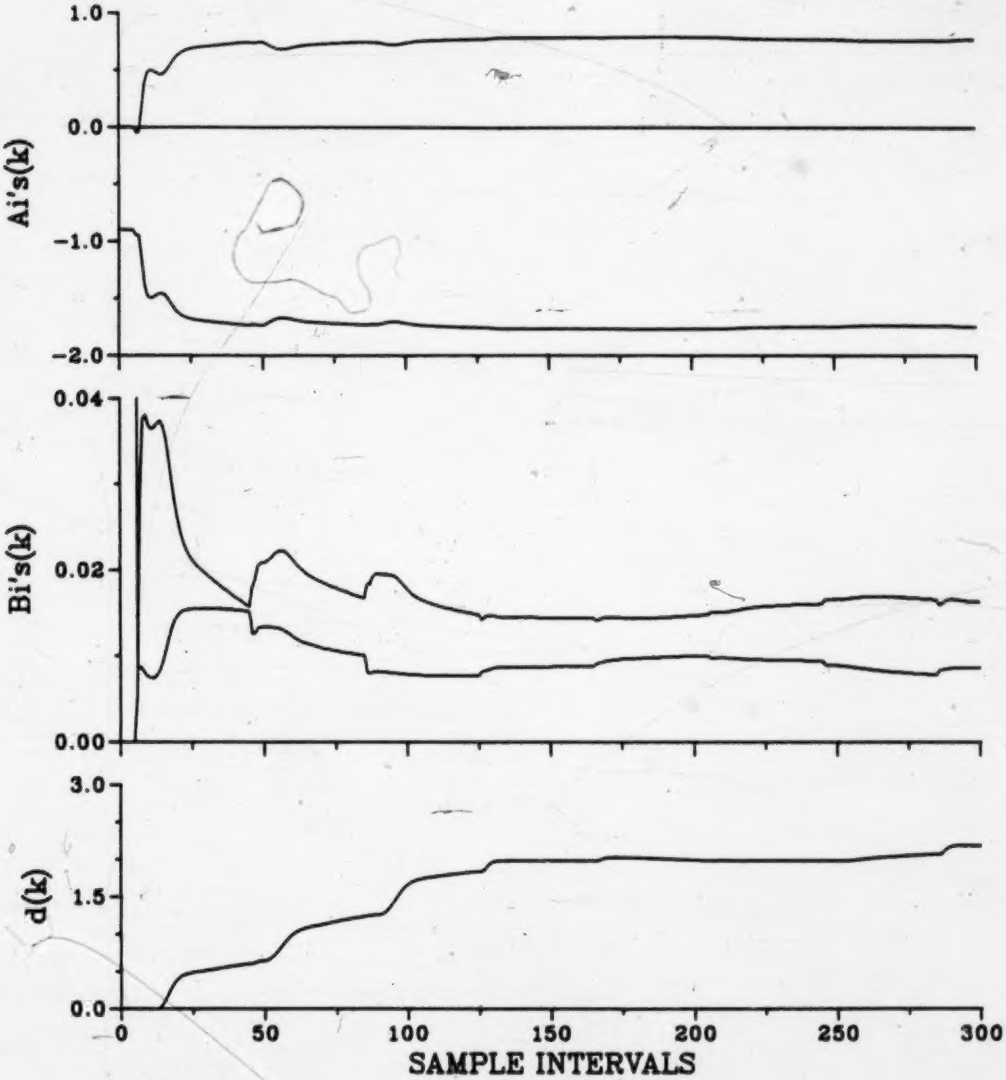

Figure 8.20b Servo self-tuning PI control of Process Model 4 (POS, P=3, M=2, C=1, $M_p$sp=15, bias=2.76)

150 sample intervals.

The regulatory capabilities of the self-tuning PID controller are summarized in Table 8.4. For PM2 the effect of the load change is removed to within a 1% deadband of the setpoint in 11.3 sample intervals. This excellent regulatory control is shown in Figure 8.21 and illustrates the fact that the self-tuning PPPID controller tuned for good servo control gives equally good regulatory control. The results for PM4 are seen in Figure 8.22. The regulatory control is sluggish but similiar to the servo control. Because the form of the PID algorithm is "setpoint-on-P&I&D", the servo and regulatory response of the self-tuning PID algorithm are similiar.

The effect of an unmeasurable load disturbance on incremental process parameter estimation is also seen in Figures 8.21 and 8.22. For PM2, the no process/model mismatch case, the effect is small. The a_i process parameters move very slightly causing a small decrease in the integral and derivative times. The movement of b_i parameter estimates is larger but does not result in a noticeable change in K_c. The closed-loop response of the process to load and setpoint changes is basically unaltered. The effect of a load change for a mismatch between process and model order is seen in Figure 8.22 for PM4. The same general trends as for PM2 are observed, but the movement of process model estimates is more pronounced. The b_i estimate movement causes a noticeable

Table 8.4 Regulatory performance of self-tuning PID control for unmeasureable load disturbances

Process Model	Mismatch	Figure No.	M_p SP	Time-Domain Response			ID Type	Comments
				M_p	t_s	IAE		
PM2	P=2 M=2 C=2	8.21	15%	59%	11.3	1.5	INC	Excellent regulatory control similiar to servo control
PM4	P=3 M=2 C=2	8.22	30%	76%	47.0	11.0	INC	Sluggish regulatory control similiar to servo control

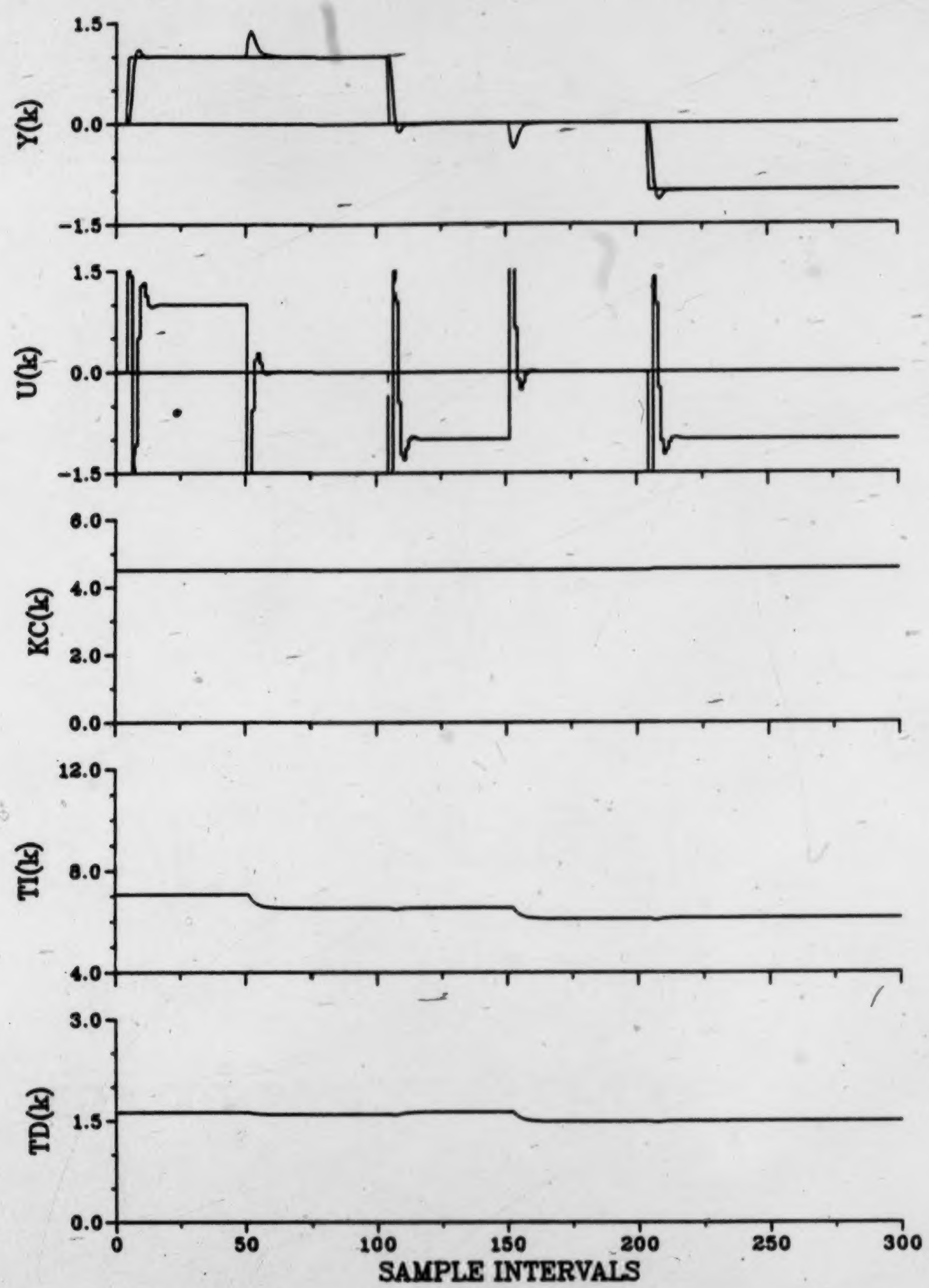

Figure 6.11a Regulatory self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15,load@50,150)

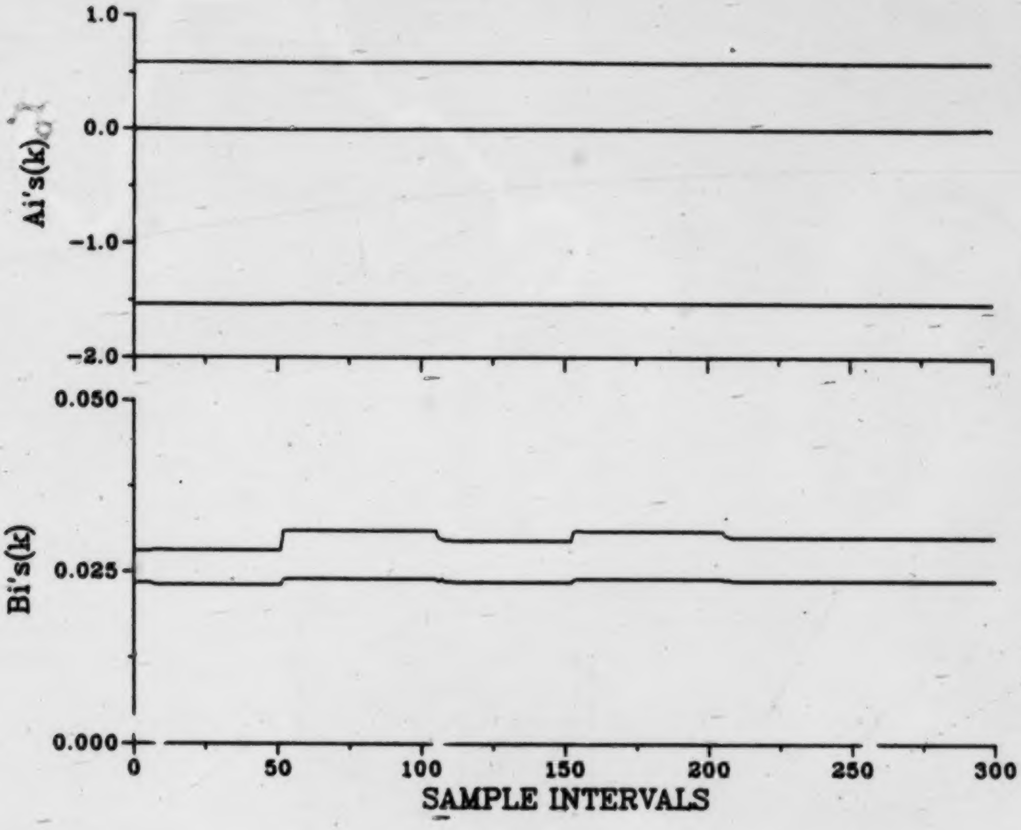

Figure 8.21b Regulatory self-tuning PID control of Process Model 2 (INC,P=M=C=2,$M_p$sp=15,load@50,150)

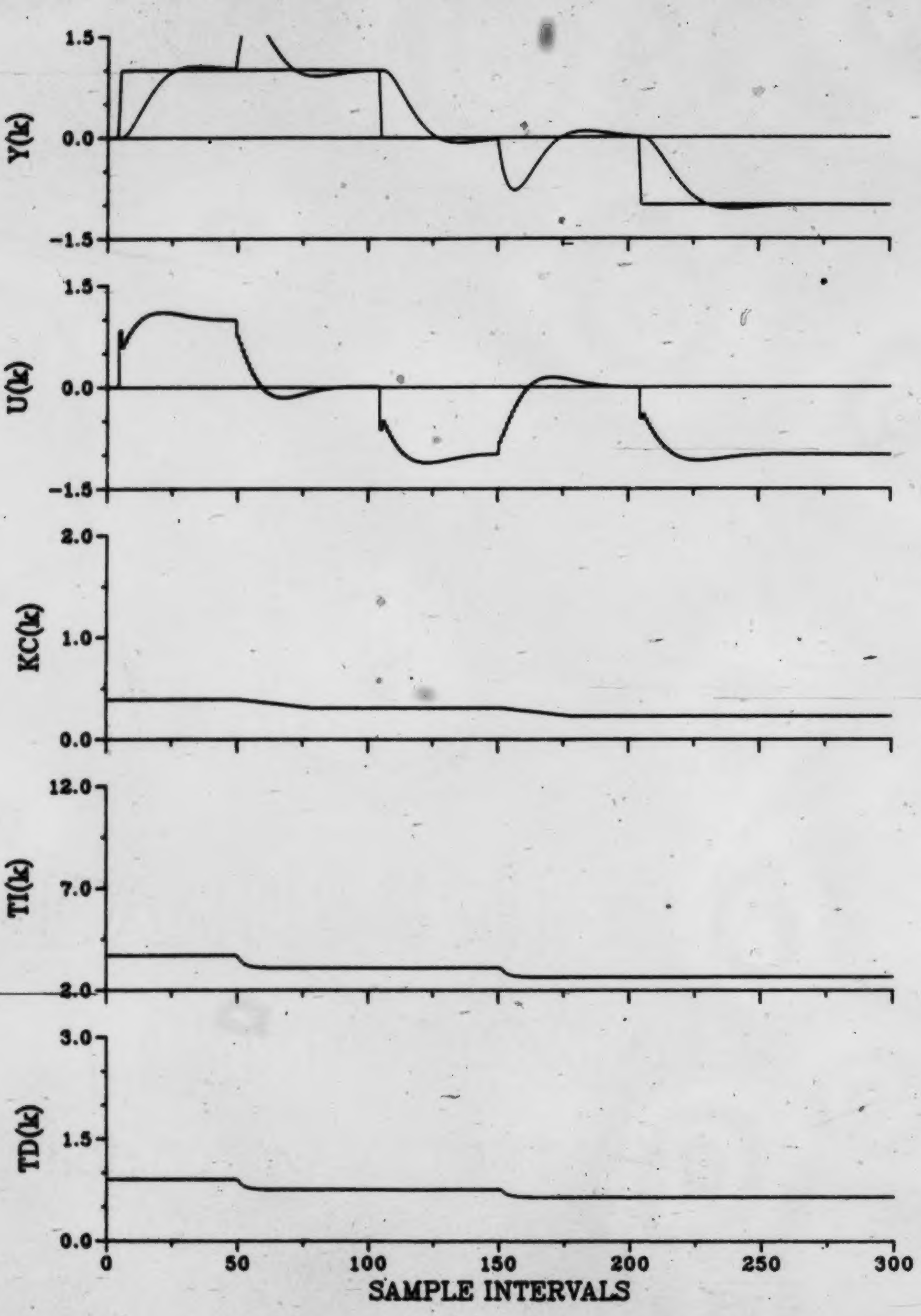

Figure 8.22a Regulatory self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=30,load@50,150)

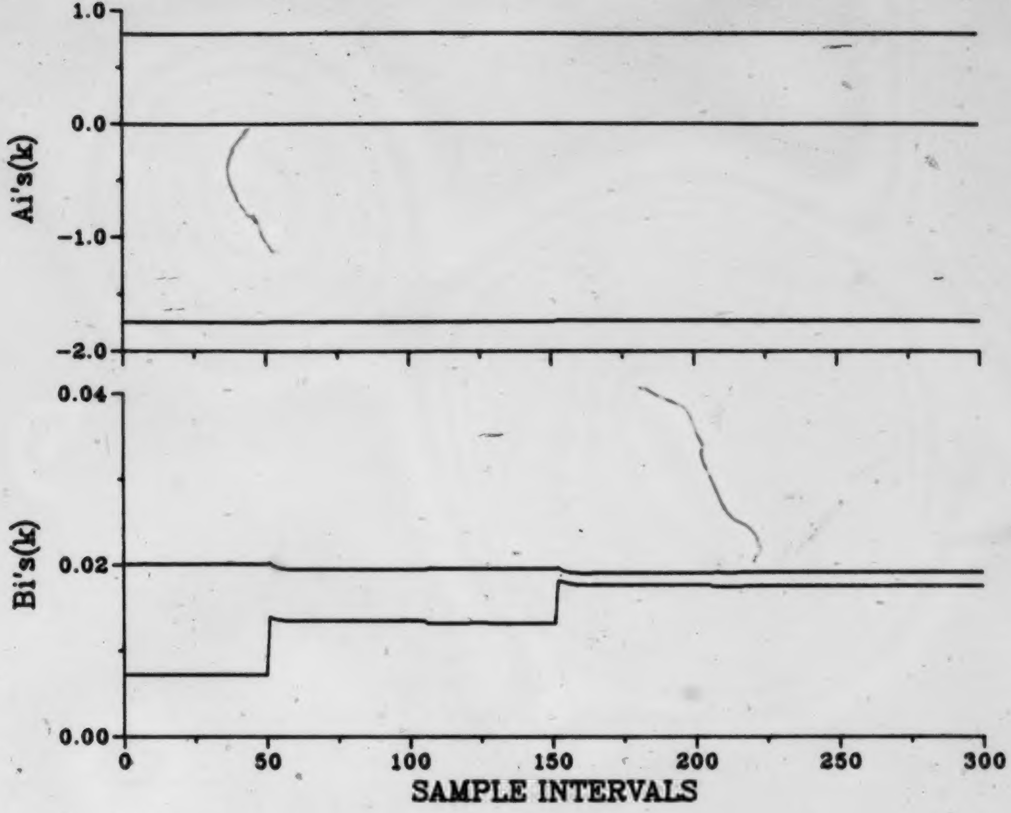

Figure 8.22b Regulatory self-tuning PID control of Process Model 4 (INC,P=3,M=C=2,$M_p$sp=30,load@50,150)

decrease in the controller gain. This translates in a marginally slower closed-loop response to setpoint and load changes.

The effect of unmeasurable load disturbances for positional estimation is much more pronounced as seen in Figures 8.23 and 8.24 for PM2 and PM4 respectively. For PM2 the a_i parameter estimates decrease to cause a large upward fluctuations in the integral time. The b_i parameter estimates decrease causing the controller gain to increase. This results in a more underdamped closed-loop response to setpoint changes. These parameter variations are also seen for PM4 in Figure 8.24. The movement of controller constants is more pronounced in the presence of process/model order mismatch. The large increase in the controller gain is the result of the decrease in the b_i parameter estimate (i.e. estimated process gain). This translates into a highly oscillatory closed-loop response to load and setpoint changes. The fluctuation of T_i and T_d track each other because T_d is reset to one-quarter of T_i if the estimated model calculates T_i to be greater than $T_i/4$.

In summary this section shows that incremental variable identification is not affected by a nonzero process bias and has a low sensitivity to unmeasured load disturbances. Positional identification is shown to be highly sensitive to both nonzero process bias and unmeasured load disturbance. It is also shown that the self-tuning PPPID algorithm tuned for servo control gives good

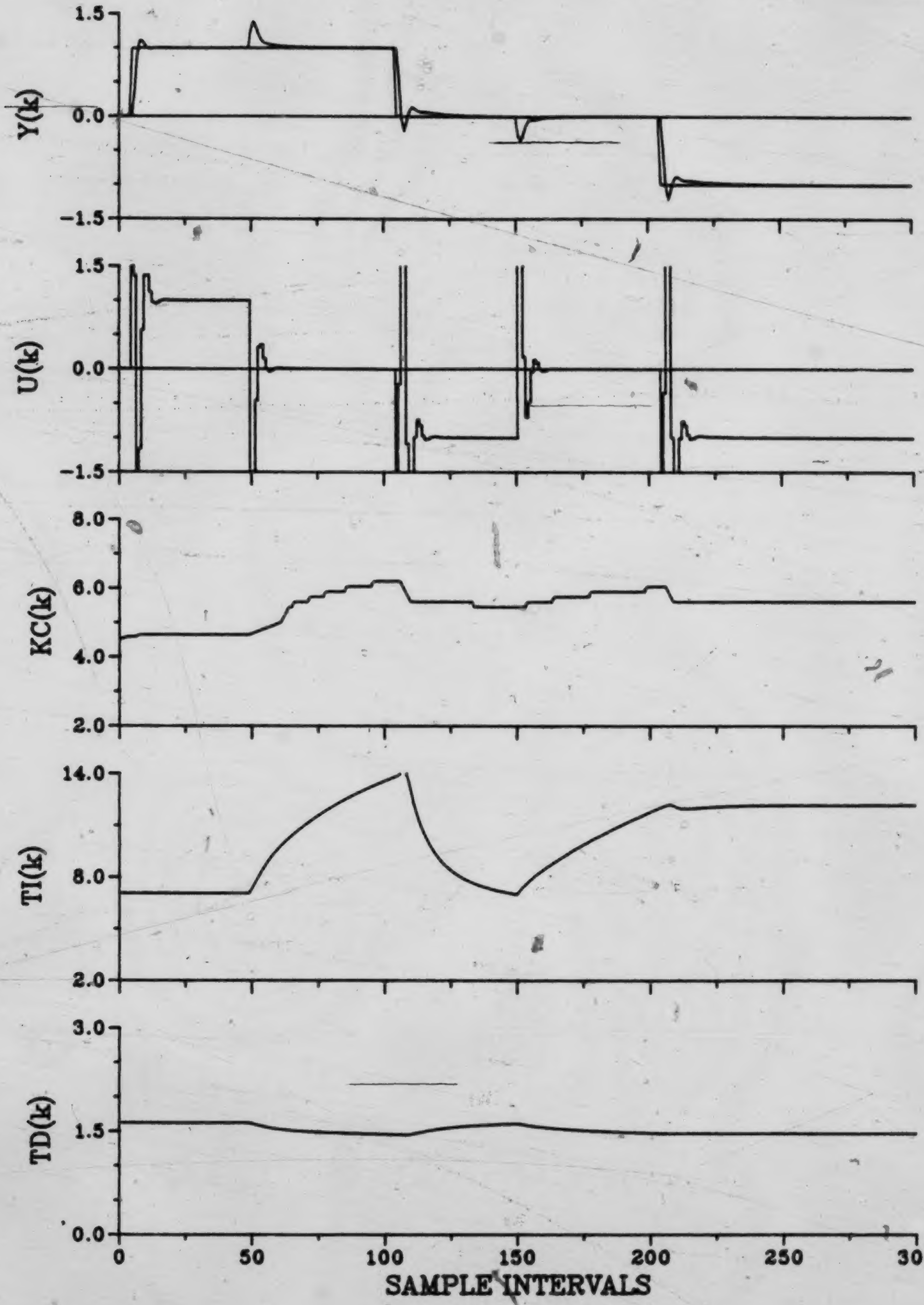

Figure 8.23a Regulatory self-tuning PID control of Process Model 2 (POS,P=M=C=2,$M_p$sp=15,load@50,150)

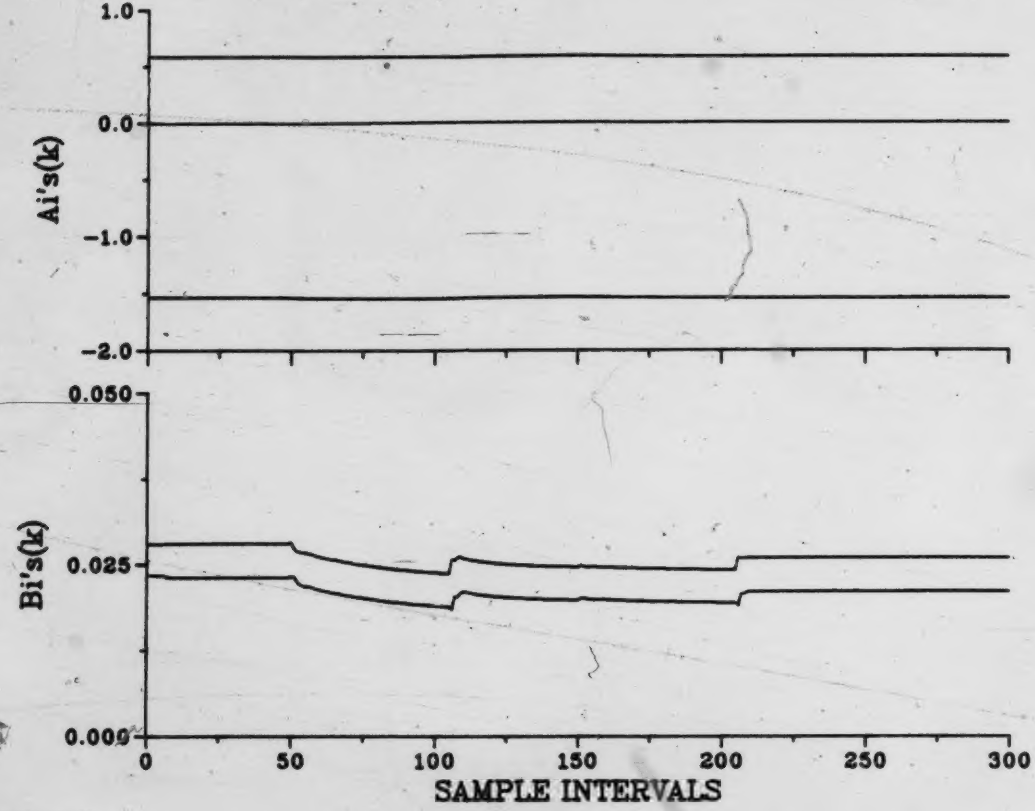

Figure 8.23b Regulatory self-tuning PID control of Process Model 2 (POS, P=M=C=2, $M_p$sp=15, load@50, 150)

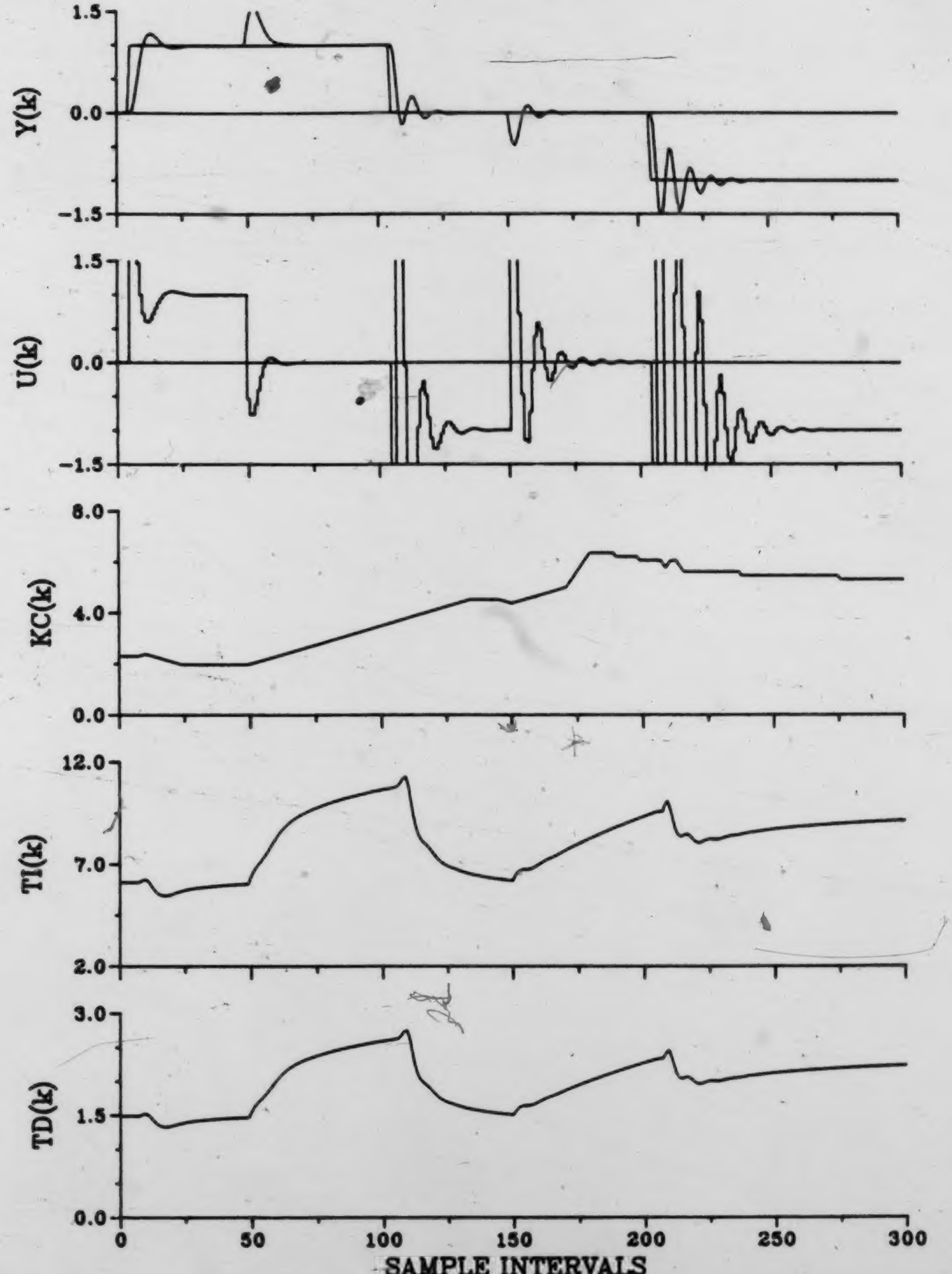

Figure 8.24a Regulatory self-tuning PID control of Process Model 4 (POS,P=3,M=C=2,$M_p$sp=30,load@50,150)

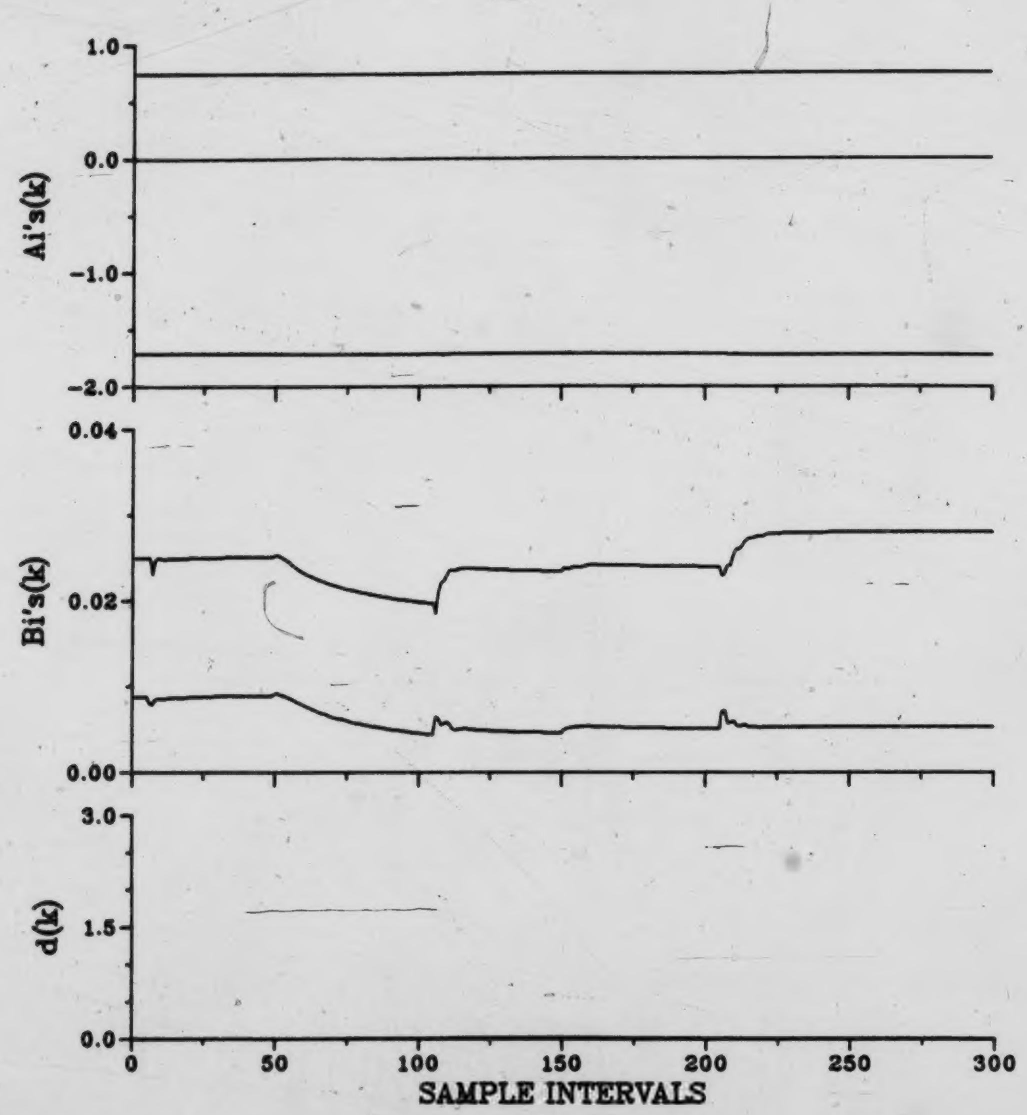

Figure 8.24b Regulatory self-tuning PID control of Process Model 4 (POS, P=3, M=C=2, $M_p$sp=30, load@50, 150)

regulatory control.

8.4 Simulation Results for Stochastic Disturbances

Stochastic disturbances are a reality which must be dealt with when implementing adaptive control strategies on industrial processes. These disturbances may loosely be called process noise and measurement noise. Process noise originates from the process itself while measurement noise results from electrical sources as the process signals are carried to the controller hardware. The objective of this set of simulation results is to evaluate the performance of the self-tuning PI(D) controller under noisy conditions. A second goal is to compare the self-tuning PI(D) control based on incremental variable identification with one based on positional identification.

The basis of the simulation runs are summarized in what follows. The PI form of the controller is used for all runs. This was done because of the sensitivity of derivative action to noisy environments. The de~~sire was to~~ examine the noise effects on the self-tuning aspect of the controller not the derivative action aspect. The noise model is:

$$N(t) = \frac{C(z^{-1})}{A(z^{-1})} \xi(t)/\Delta \qquad (8.2)$$

where $N(t)$ and $\xi(t)$ are coloured and white noise sequences respectively, $A(z^{-1})$ and $C(z^{-1})$ are monic polynomials in the backward shift operator z^{-1} and Δ is $(1-z^{-1})$. The particular choice of $C(z^{-1})$ used for these results is:

$$C(z^{-1}) = 1 - 0.8z^{-1} \tag{8.3}$$

This choice results in the noise being correlated with the estimation algorithm input-output data regressor. The $A(z^{-1})$ polynomial is the denominator polynomial of the process model. Process Model 1 and Process Model 4 are chosen as the ideal and the mismatch cases respectively. The estimation method is recursive extended least squares (RELS) with one c_i parameter estimated.

The simulation results for PM1 are shown in Figure 8.25 for the self-tuning PI controller based on incremental variable identification for the ideal case. The variance of the white noise input, $\xi(t)$ is σ^2=0.00111. The servo performance of the controller is excellent with the process parameter estimates and the controller constants being very close to the results of the equivalent no noise case (see Figure 7.1). It is interesting to note that the convergence of the c_1 parameter estimate is slow and the nonstationary nature of the disturbance as seen by the drifting controller output signal.

The self-tuning PI control results for PM4 is seen in Figure 8.26. These results show how incremental identification is affected by both noise and mismatch between process and model order. The variance of the white noise sequence is σ^2=0.000025. The closed-loop response is

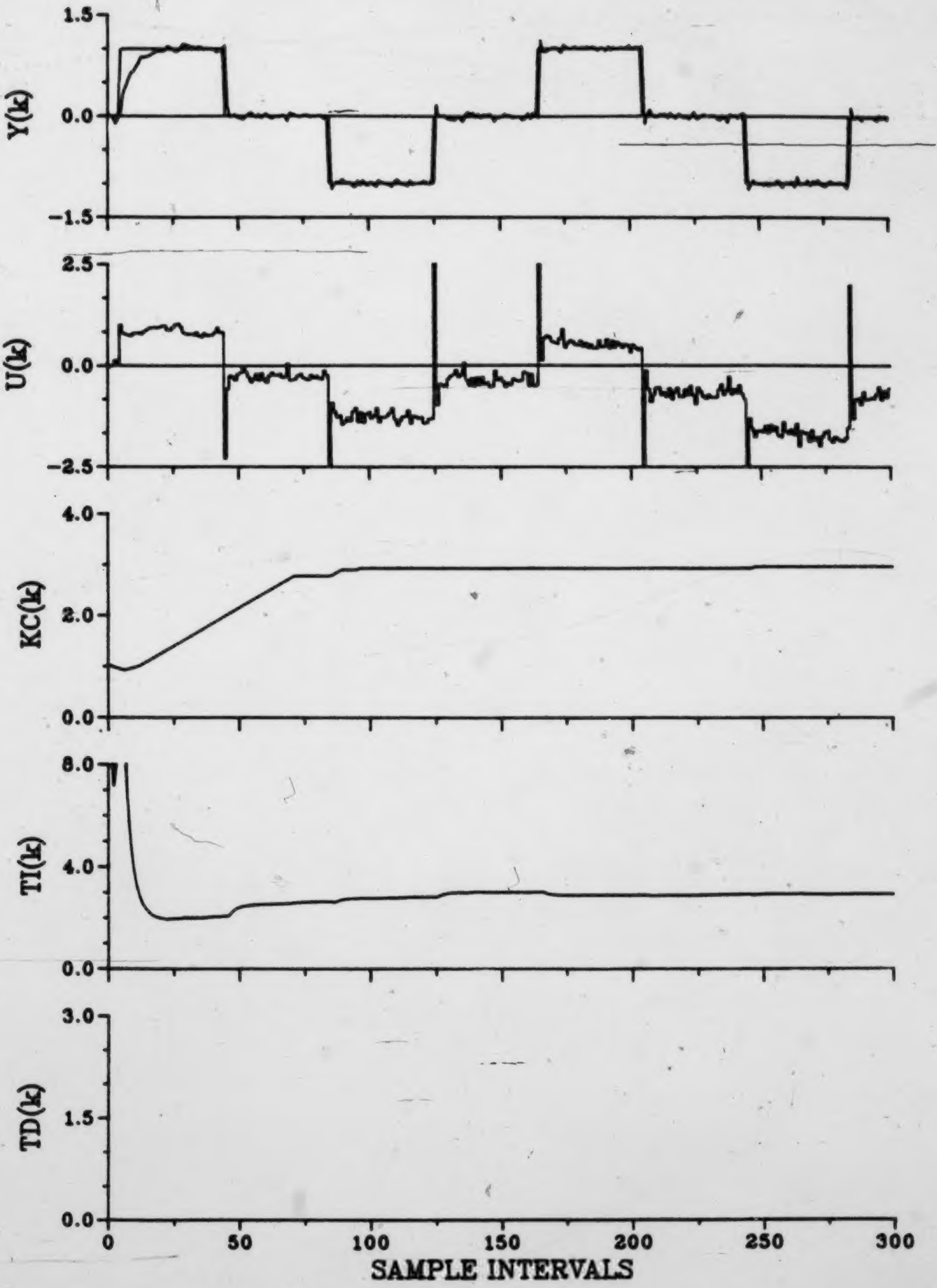

Figure 8.25a Servo self-tuning PI control of Process Model 1 (INC,P=M=C=1,$M_p$sp=15,noise)

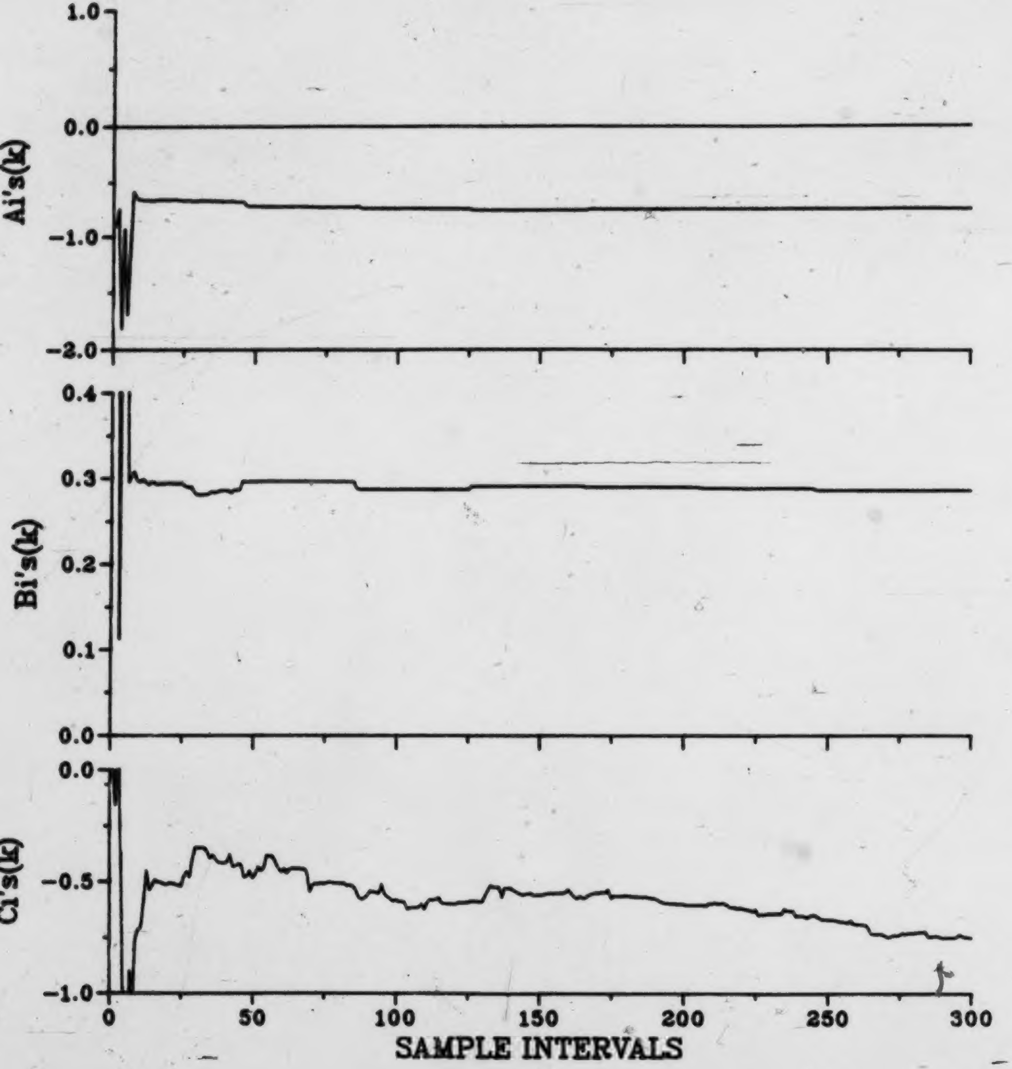

Figure 8.25b Servo self-tuning PI control of Process Model 1 (INC,P=M=C=1,$M_p$sp=15,noise)

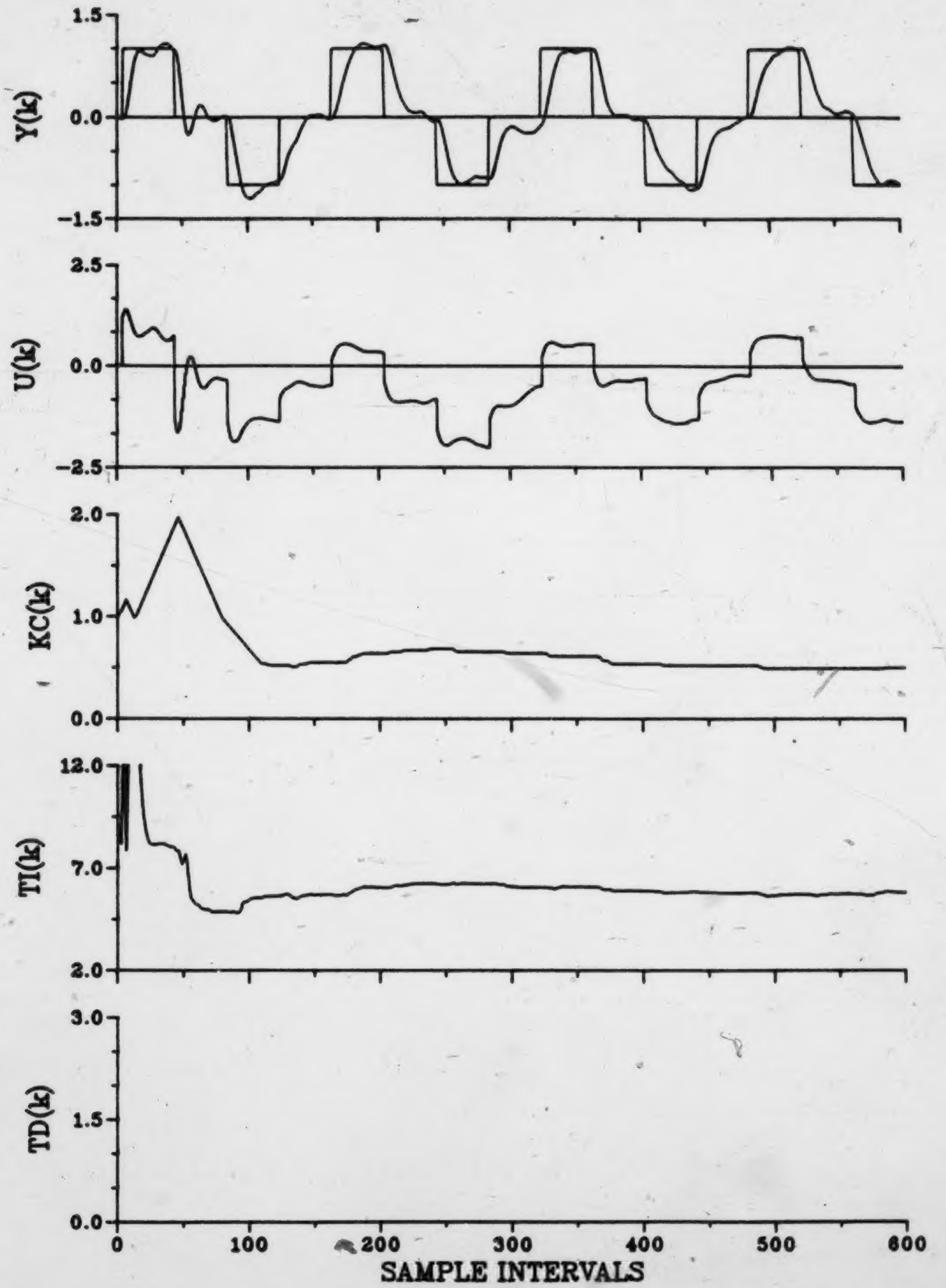

Figure 8.26a Servo self-tuning PI control of Process Model 4 (INC,P=3,M=2,C=1,$M_p$sp=15,noise)

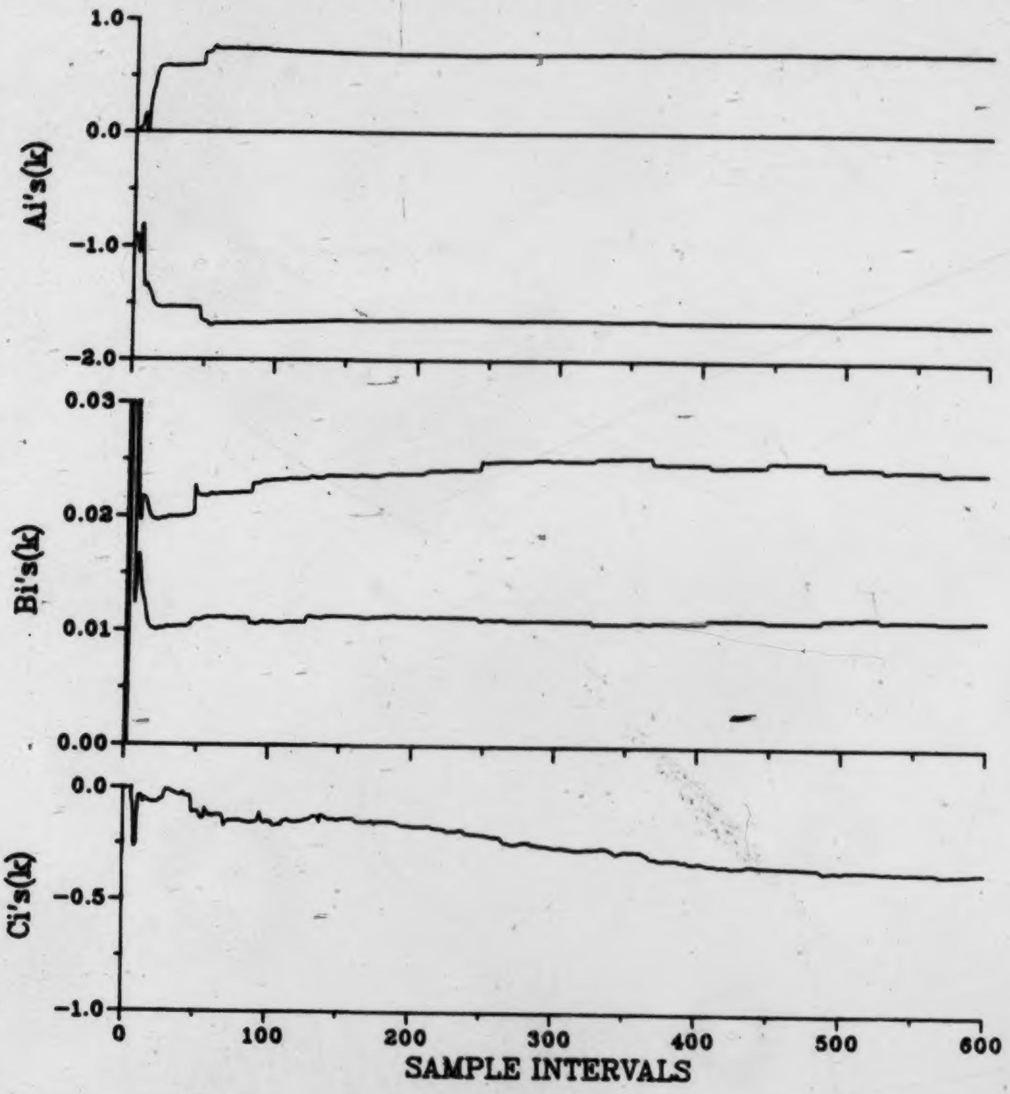

Figure 8.26b Servo self-tuning PI control of Process Model 4 (INC, $P=3, M=2, C=1, M_p sp=15$, noise)

good, similiar to the equivalent no-noise case (see Figure 7.8). The process parameter estimates are slightly biased from the no-noise case.

In summary it is evident that the self-tuning PPPI controller can handle stochastic disturbances if there is no mismatch between the process and the model. It is, however, sensitive to noise in the presence of mismatch. In this situation the input and output signals should be filtered through an analog filter before being passed to the process parameter estimation algorithm.

8.5 Summary

The self-tuning PPPI(D) control law has now been evaluated in the presence of time delays, process bias, load disturbances and noise in addition to the process/model order mismatch results of Chapter 7. As a result of these simulation results the following additional conclusions are drawn and summarized in point form below:

1. The performance of the self-tuning PID controller is found to be good for a known time delay. The control design method incorporated the time delay into the controller gain calculation and detuned the controller gain. Incremental variable identification for a mismatch of time delay and model order results in poor control. An overestimation of the time delay with estimation of extra b_i parameters improves the performance. Time delay compensation is shown to

result in good control for a known or unknown delay.

2. The self-tuning PID controller tuned for servo response gives good regulatory control for load disturbances.
3. The self-tuning PID controller based on incremental identification is found to have a low sensitivity to process bias and deterministic load disturbances whereas positional identification is very sensitive to both process bias and load changes.
4. Good control is achieved in the presence of stochastic disturbances for no mismatch of process and model order. With mismatch degradation of control performance is seen because incremental identification is sensitive to the additive effect of the high frequency dynamics of model order mismatch and noise. Positional identification is less sensitive to these effects.

The feasibility of the controller design strategy and the capability of self-tuning PPPI(D) controller has been demonstrated for linear processes under various environments. It is projected that it is also capable of controlling more difficult processes characterizsed by nonlinearities and measurement noise in addition to the issues considered in this chapter. However, the sensitivity of the self-tuning controller based on incremental identification should be addressed. Proper sampling rate choice must be made to reduce the mismatch effects of high frequency dynamics. The effect of noise in incremental identification leads to increased parameter variance. This can be attenuated by proper analog filtering.

9. Performance of Self-Tuning PI Controller on a Simulated Nonlinear System

9.1 Introduction

The conclusions drawn in Chapters 7 and 8 for the self-tuning PI(D) controller are based on simulation studies using linear transfer function process models. The objective of this chapter is to evaluate the performance of the self-tuning PI(D) controller on a highly nonlinear system which approaches the complexity of a real process. The particular system chosen is the distillation process, an important separation method in the refining and chemical industries. The self-tuning controller, based on the pole placement design principle and incremental parameter estimation, is examined under the combined effects of time delays, noise and nonlinearities. The initialization parameters for the controller are summarized in Table 7.1 with the exception of two variable forgetting factor parameters. The initial and minimum forgetting factor are chosen to be 0.998 instead of 0.99.

Section 9.2 describes the nonlinear distillation column simulation package. SISO control of the overhead and bottoms composition (but not simultaneously) are considered in Sections 9.3 and 9.4 respectively. Section 9.5 summarizes the results of this chapter.

9.2 Distillation Column Simulator

The distillation column simulation package (DCSP) employed for these studies is a simulator developed by Kan (1982) and modified by Nazer (1981). It is a dynamic, nonlinear model of a binary distillation column based on both material and energy balances equation. Assumptions made in the derivation of the model are:

1. constant liquid mass holdup,
2. negligible vapor mass holdup,
3. perfect mixing at each stage,
4. instantaneous heat and mass transfer equilibrium.

A detailed description of the simulator is given by Kan (1982).

For the purposes of this study the distillation column model is configured to make the binary separation of methanol and water. Appendix A contains a sample listing of the data file used to initialize the simulator for SISO control of the overhead composition. The majority of the data file contains thermodynamic data such as enthalpies, flow rates and compositions. Lines 1,7,8 and 10 are the only lines modified for the simulation results presented in this chapter. Line 1 specifies the option of servo control of the top or bottom loop or regulatory control in the presence of feed flow rate load changes. Lines 7 and 8 determine the time and magnitude of setpoint changes or load disturbances. Line 10 specifies the amount of measurement noise added to the control, output and load signals at every control interval. All simulation runs are SISO runs. No attempt was made to study the multi-loop performance of the self-tuning PI controller.

9.3 Overhead Composition Control

Control of the overhead composition is achieved by manipulating the reflux flow rate of the simulated column. The column feed and reboiler steam flow rates are held constant. A time series analysis (TSA) (Vien, 1986) of closed-loop input-output data was completed to determine the model order of the discrete transfer function relating the reflux flow rate and the overhead composition. The results showed that the model was first order with one a_i and one b_i parameter. Initial model parameter estimation was then completed using the nonrecursive nonlinear least squares option of Vien's Process Identification Package (PIP) (Vien, 1986). The resultant model for T_s=3 minutes is estimated to be:

$$G(z^{-1}) = \frac{0.537z^{-1}}{1 - 0.351z^{-1}} \tag{9.1}$$

Because the time series analysis shows that the model order of the overhead loop is first order, all simulations are done with a self-tuning PI controller based on a first order model. Due to the presence of noise and nonlinearities, a recursive extended least squares estimation algorithm is used with one c_i estimated.

The servo self-tuning control results are summarized in Table 9.1 for two sampling rates for the no noise and the noise cases. All results presented in this chapter are based on incremental variable identification. Figure 9.1 shows the no noise results for T_s=3. Closed-loop response to setpoint changes is good. The change in setpoint from 95% to 94% at T=305 minutes results in an oscillatory response. This change also causes a distinct change in the process parameter estimates and the controller gain. Convergence of the process parameter estimates is good but slight shifts are seen at setpoint changes as the process output moves above and below the mean level. This is partially attributed to the nonlinearity of the process. The estimated deterministic process model at T=800 minutes is:

$$G(z^{-1}) = \frac{0.586z^{-1}}{1 - 0.396z^{-1}} \tag{9.2}$$

The addition of measurement noise, as seen in Figure 9.2, results in biased parameter estimates. The estimated deterministic process model at T=800 minutes is now:

$$G(z^{-1}) = \frac{0.661z^{-1}}{1 - 0.421z^{-1}} \tag{9.3}$$

However, the resultant control is good in spite of biased parameters.

The effect of choosing T_s=1 minute is shown in Figures

Table 9.1 Servo self-tuning PI control of overhead and bottoms composition of distillation column simulator (DCSP)

Run	Sample Interval	Figure No.	Noise	IAE	Comments
DCSPTL.3.D00	3.0	9.1	No	27.7	Good servo control
DCSPTL.3.D01	3.0	9.2	Yes	112.0	Good servo control, some biasing of parameter estimates
DCSPTL.1.D00	1.0	9.3	No	10.1	Good servo control
DCSPTL.1.D01	1.0	9.4	Yes	83.2	Good servo control, severe biasing of parameter estimates
DCSPTL.3.D05	3.0	9.5	Yes	97.4	Good regulatory control
DCSPBL.3.D02	3.0	9.6	No	79.4	Good underdamped servo control
DCSPBL.3.D03	3.0	9.7	Yes	130.0	Reasonable underdamped servo control
DCSPBL.3.D06	3.0	9.8	Yes	496.0	Acceptable regulatory control

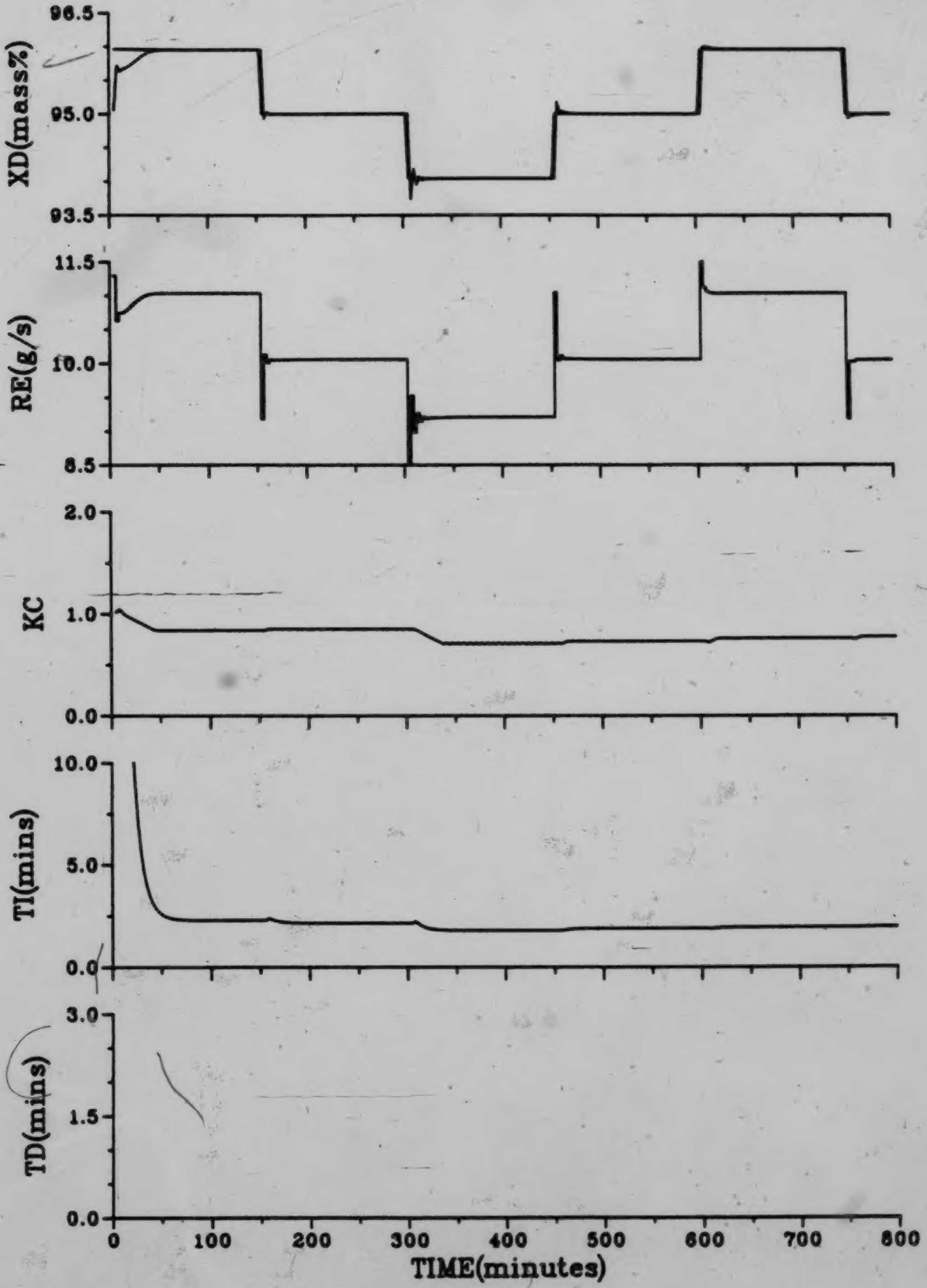

Figure 9.1a Servo self-tuning PI control of OVHD composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,no noise)

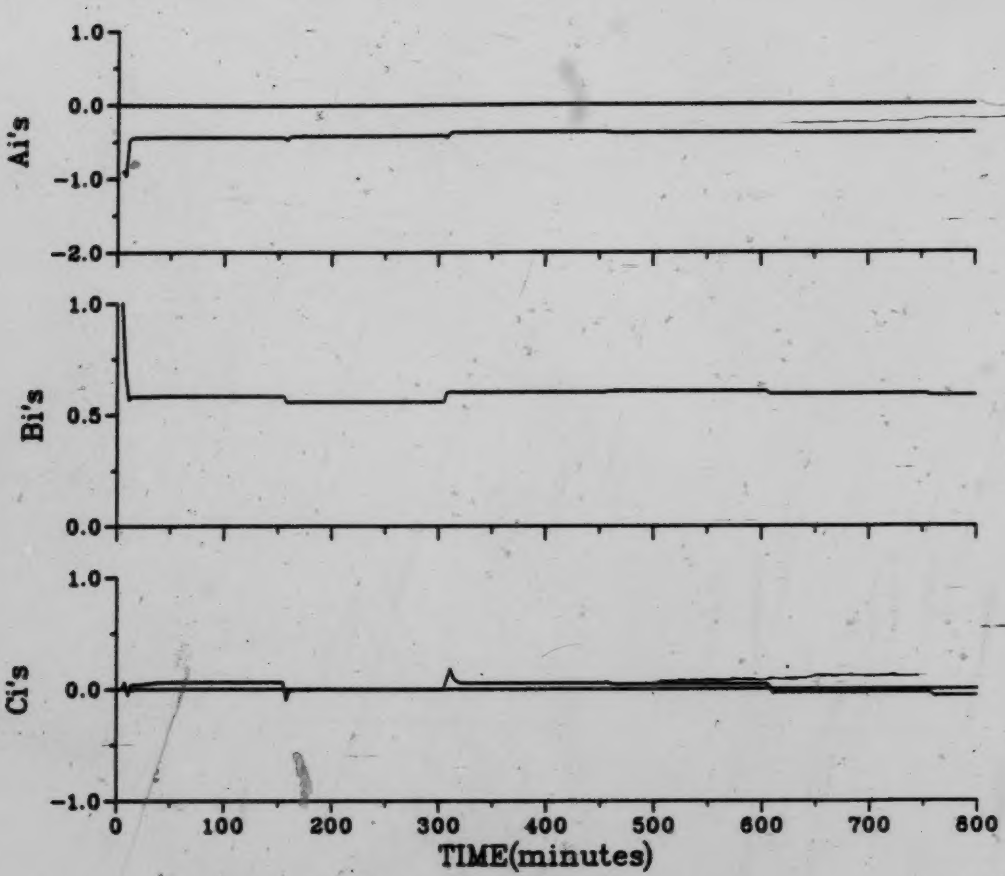

Figure 9.1b Servo self-tuning PI control of OVHD composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,no noise)

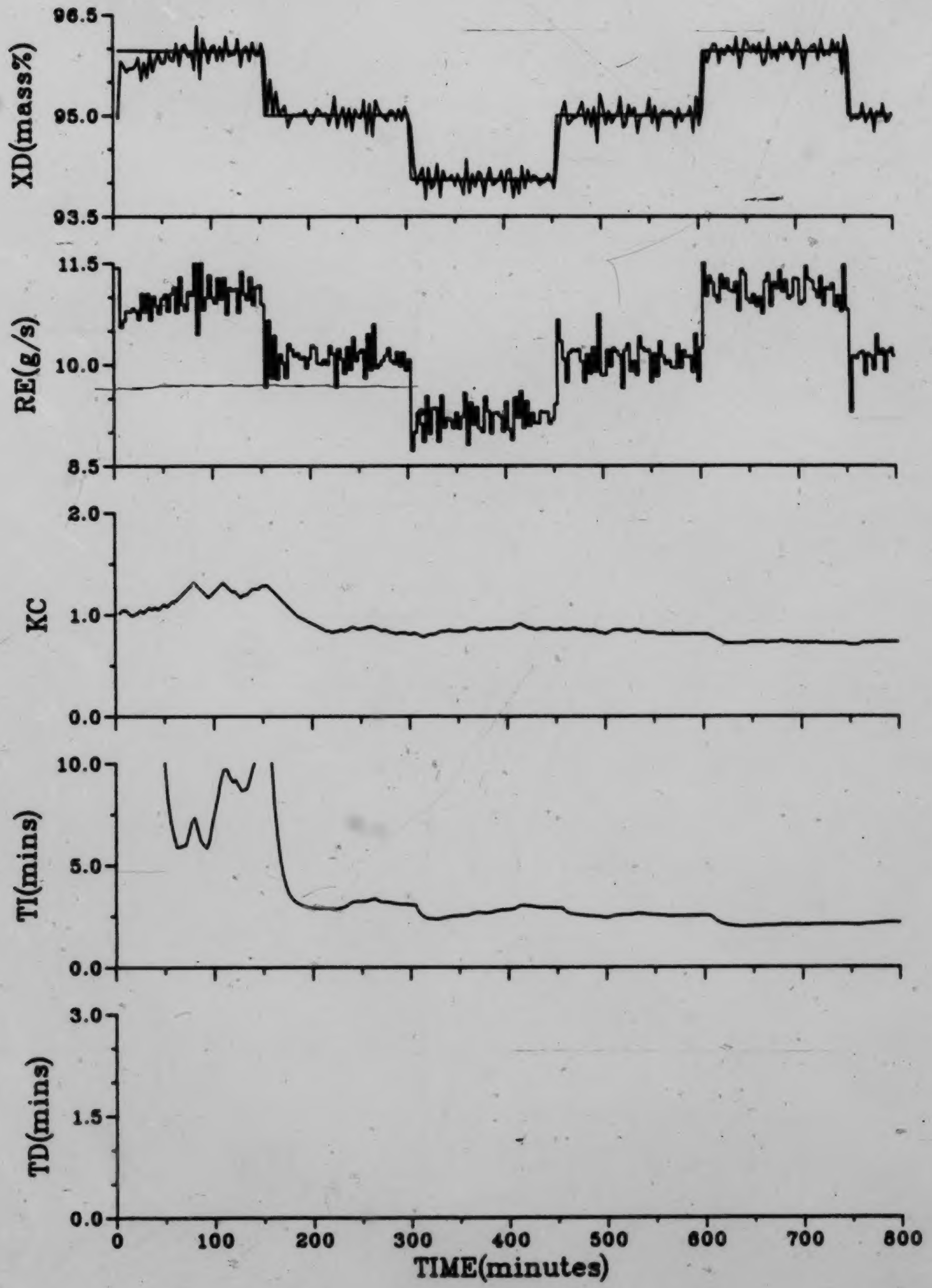

Figure 9.2a Servo self-tuning PI control of OVHD composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,noise)

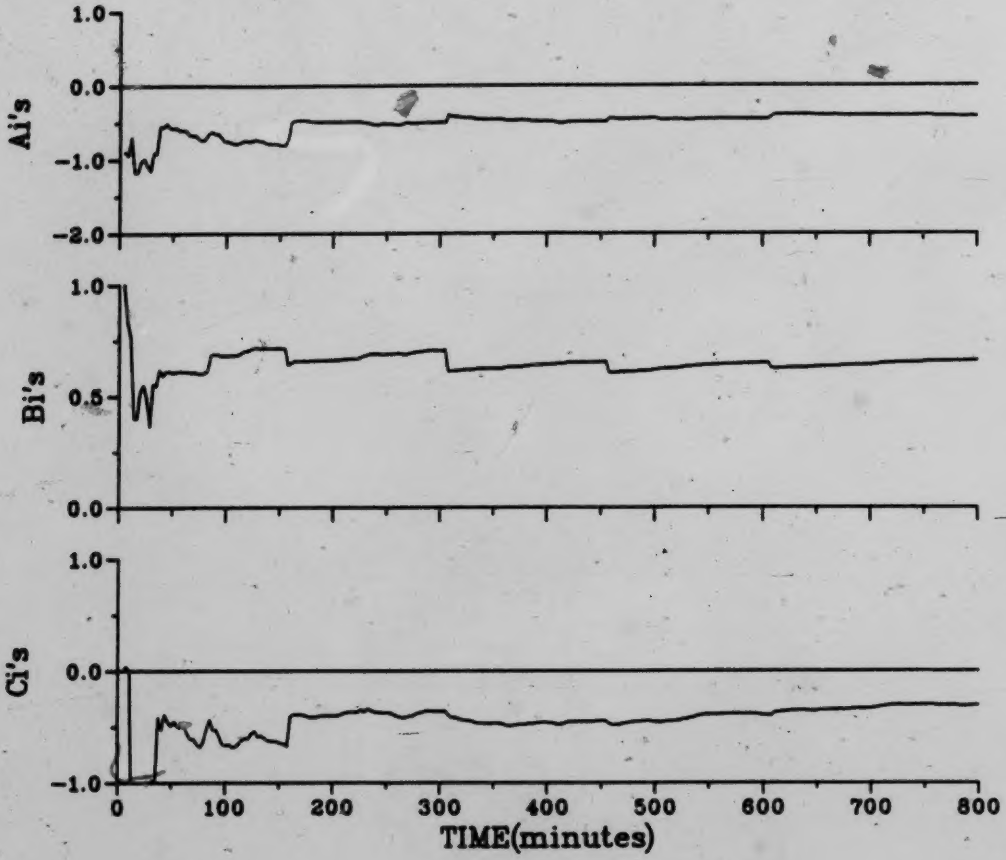

Figure 9.2b Servo self-tuning PI control of OVHD composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,noise)

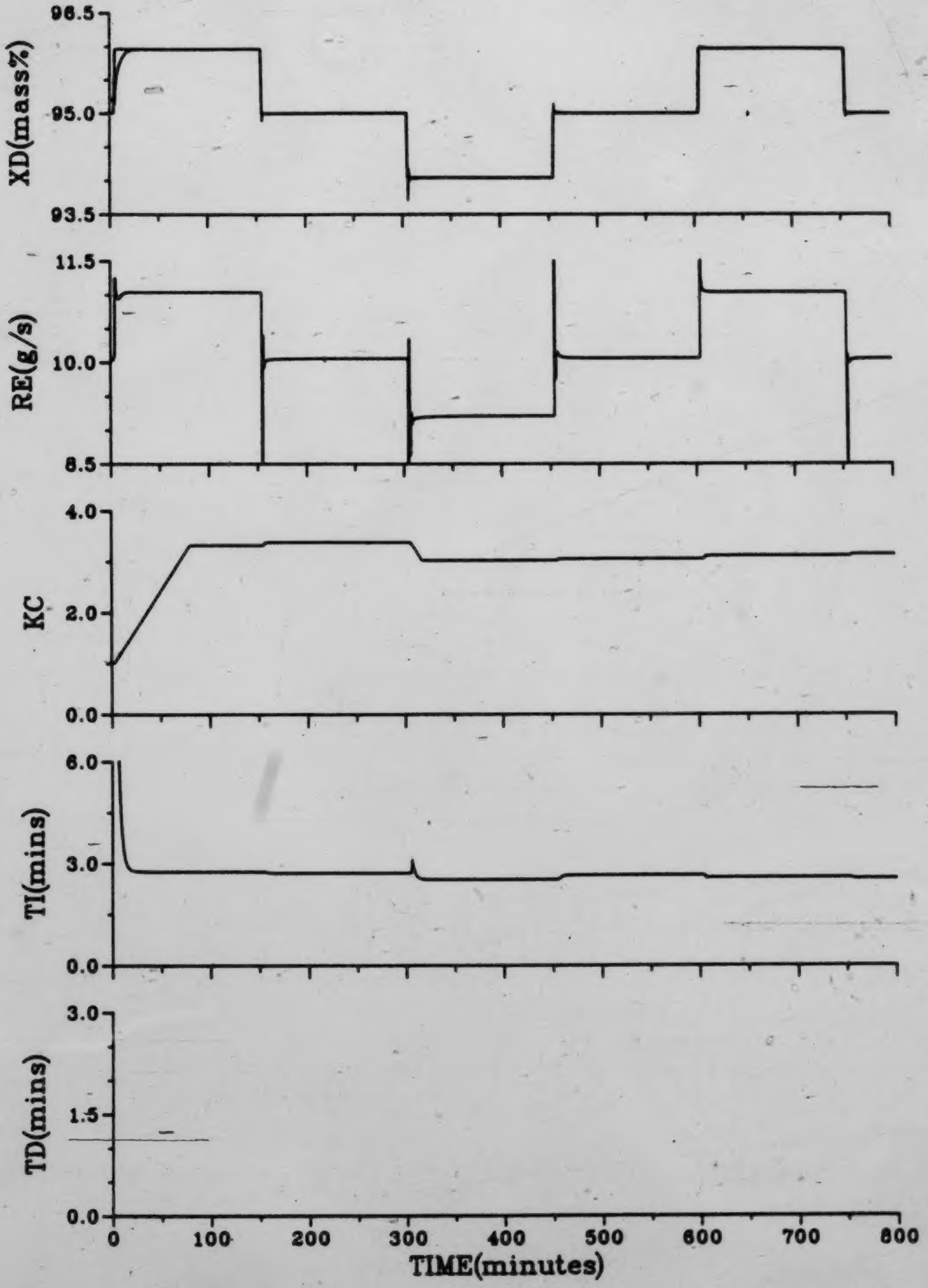

Figure 9.3 Servo self-tuning PI control of OVHD composition (INC,P=M=C=1,$M_p$sp=15,T_s=1,no noise)

9.3 and 9.4. For the no noise case closed-loop control is excellent. Again the most oscillatory response is seen at T=305 minutes as the setpoint moves to 94% from 95%. The estimated process model at T=800 minutes is:

$$G(z^{-1}) = \frac{0.259z^{-1}}{1 - 0.718z^{-1}} \qquad (9.4)$$

The effects of measurement noise for $T_s=1$ are shown in Figure 9.4. The estimated process model at T=800 minutes is:

$$G(z^{-1}) = \frac{0.486z^{-1}}{1 - 0.494z^{-1}} \qquad (9.5)$$

A severe biasing of the estimate process parameters appears when compared with the no noise model (cf. Equation 9.4). This is attributed to the additive effects of the noise implemented in DCSP and the sensitivity of incremental variable identification to high frequency noise. Measurement noise will cause biased process parameter estimates if there is a mismatch between the estimated noise model and the true noise model, as is the case here. At the higher sampling rate $(T_s=1)$ biasing effects of the noise are magnified and incremental variable identification estimates a model which incorporates the high frequency noise dynamics. The biased process parameter estimates result in a reduction of the calculated K_c and T_i of the controller. Even with this biasing, the controller

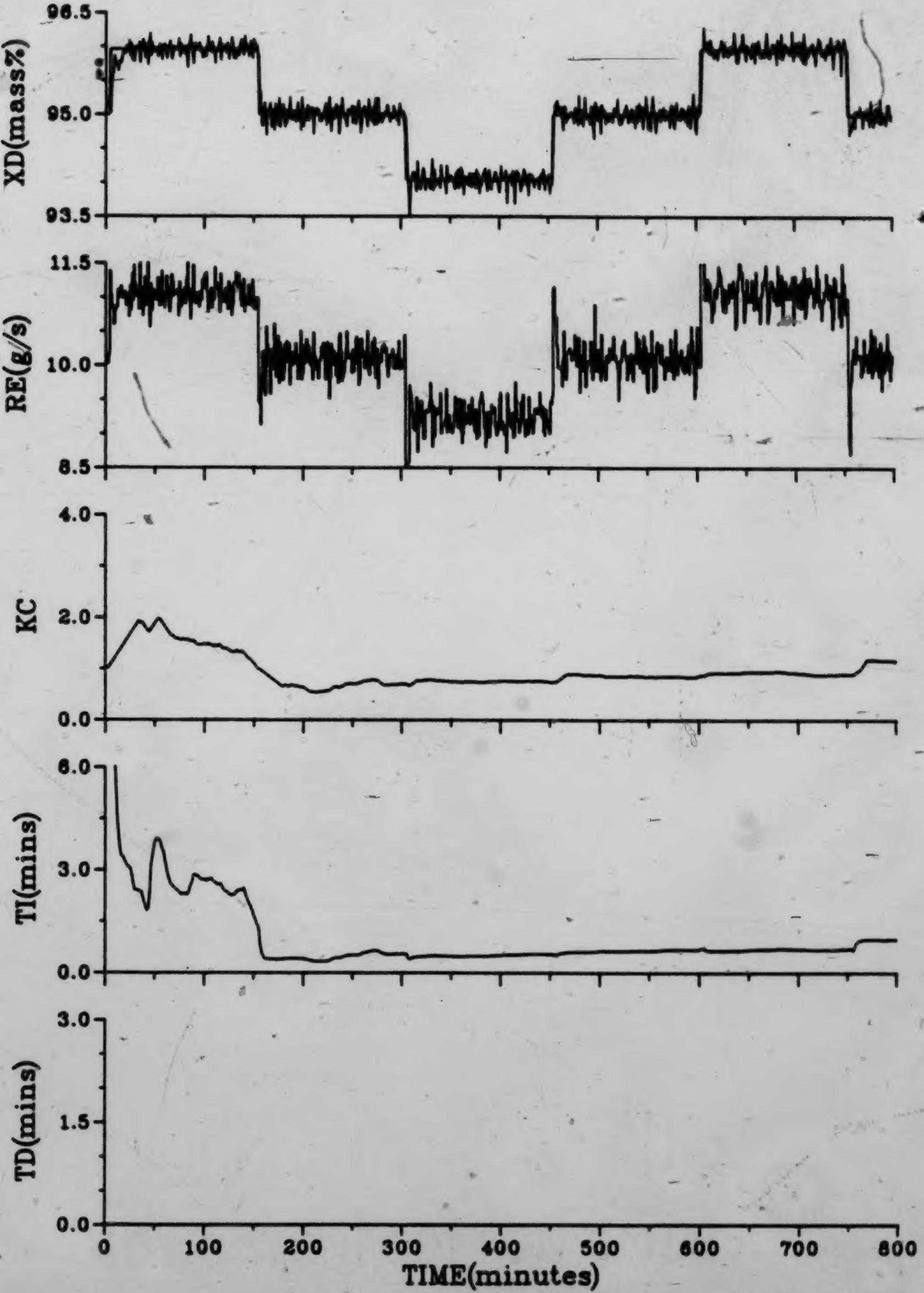

Figure 9.4a Servo self-tuning PI control of OVHD composition (INC,P=M=C=1,$M_p$sp=15,T_s=1,noise)

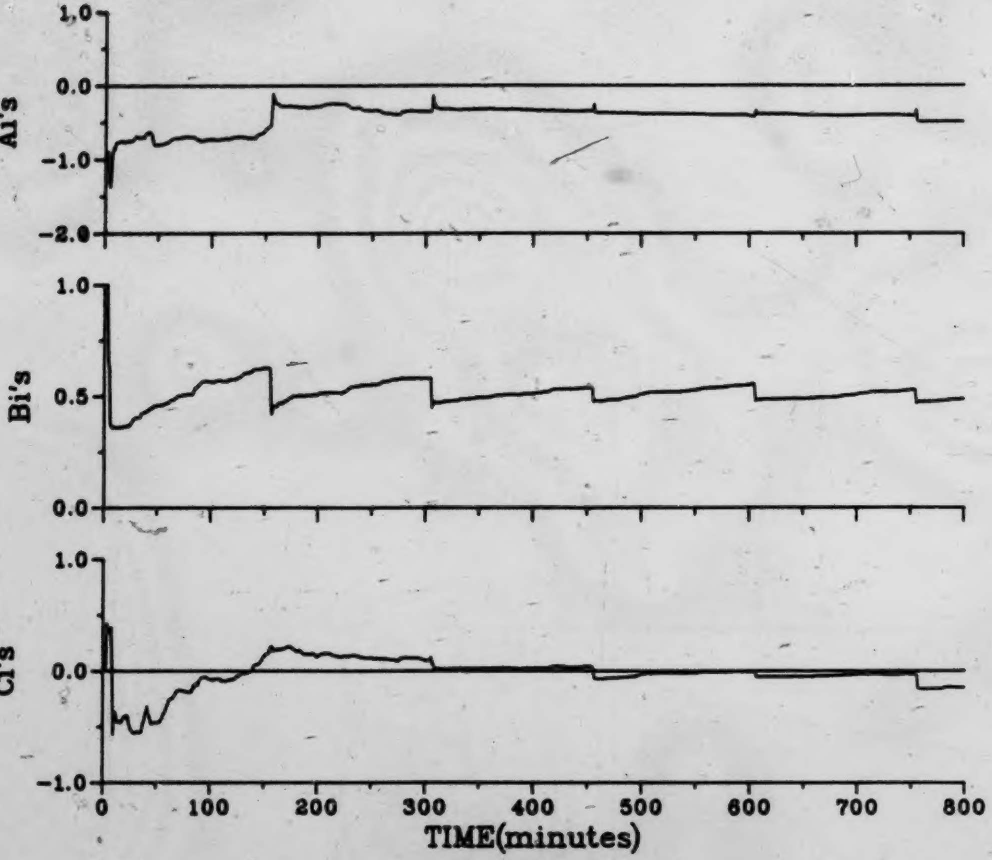

Figure 9.4b Servo self-tuning PI control of OVHD composition (INC, P=M=C=1, $M_p$sp=15, T_s=1, noise)

performance is judged to be good based upon the integral of the absolute error (IAE) calculation.

The regulatory control ability of the self-tuning PPPI controller tuned for servo responses is demonstrated in Figure 9.5. The unmeasurable load disturbances in this case are a ±25% change in feed flow rate. The controller is operated with the identification off. The controller parameters, K_c and T_i, are averaged from the results of Figure 9.2 and are K_c=0.793 and T_i=2.63. Regulatory control is excellent. The nonlinear nature of the distillation column simulator is demonstrated at the -25% change in feed flow rate. The reflux flow rate moves sharply upward at this point. No noticeable change in reflux flow rates is seen for a +25% change in feed flow rate.

9.4 Bottoms Composition Control

Control of the bottoms composition of the distillation column simulator is achieved by manipulating the steam flow rate to the reboiler. For this scheme the feed and reflux flow rates are held constant. A times series analysis (TSA) of closed-loop input and output data showed the order of the model relating the steam flow rate to the bottoms composition to be first order. A first order model fit for T_s=3 minutes using a nonrecursive nonlinear least squares estimation algorithm resulted in the following model:

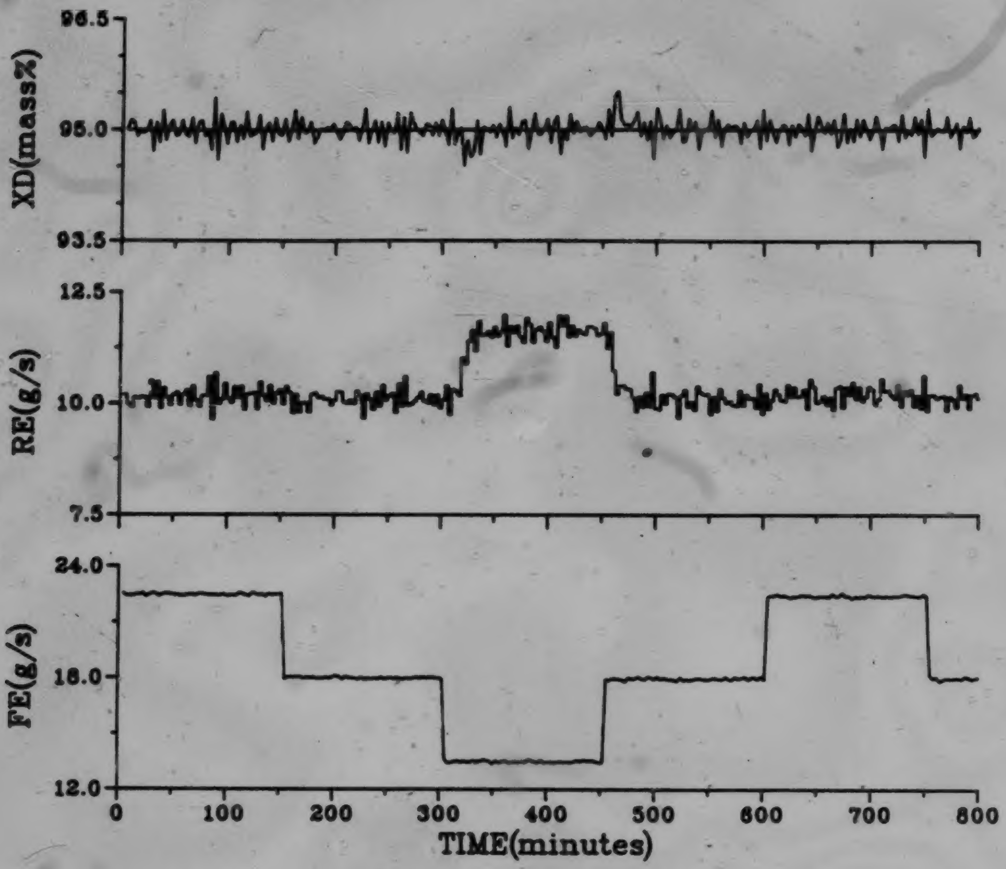

Figure 9.5 Regulatory PI control of OVHD composition (K_c=0.793,T_i=2.63,T_s=3,noise)

$$G(z^{-1}) = \frac{-1.245z^{-1}}{1 - 0.831z^{-1}} \tag{9.6}$$

These results suggested that a self-tuning PI controller be implemented.

The choice of the sampling period for the bottom loop is dictated by the desire that DCSP emulate the experimental column conditions as closely as possible. The bottoms composition of the experimental column is analyzed by a gas chromatograph which has a cycle time of 3 minutes. Therefore, a reasonable choice of T_s is 3 minutes. This introduces a time delay of one sample interval into the bottom loop.

The servo self-tuning PI runs for control of the bottoms composition are summarized in Table 9.1. Figure 9.6 describes the tuning phase and the resultant closed-loop response for no noise added to the simulator. Process parameter estimates converge quickly. The estimated deterministic process model at T=800 minutes for recursive extended least squares is:

$$G(z^{-1}) = \frac{-1.314z^{-1}}{1 - 0.869z^{-1}} \tag{9.7}$$

The closed-loop performance for setpoint changes in the bottoms composition is good with a peak overshoot as specified (i.e. M_p=15%). The effect of noise on the bottom loop is shown Figure 9.7. The most significant result is

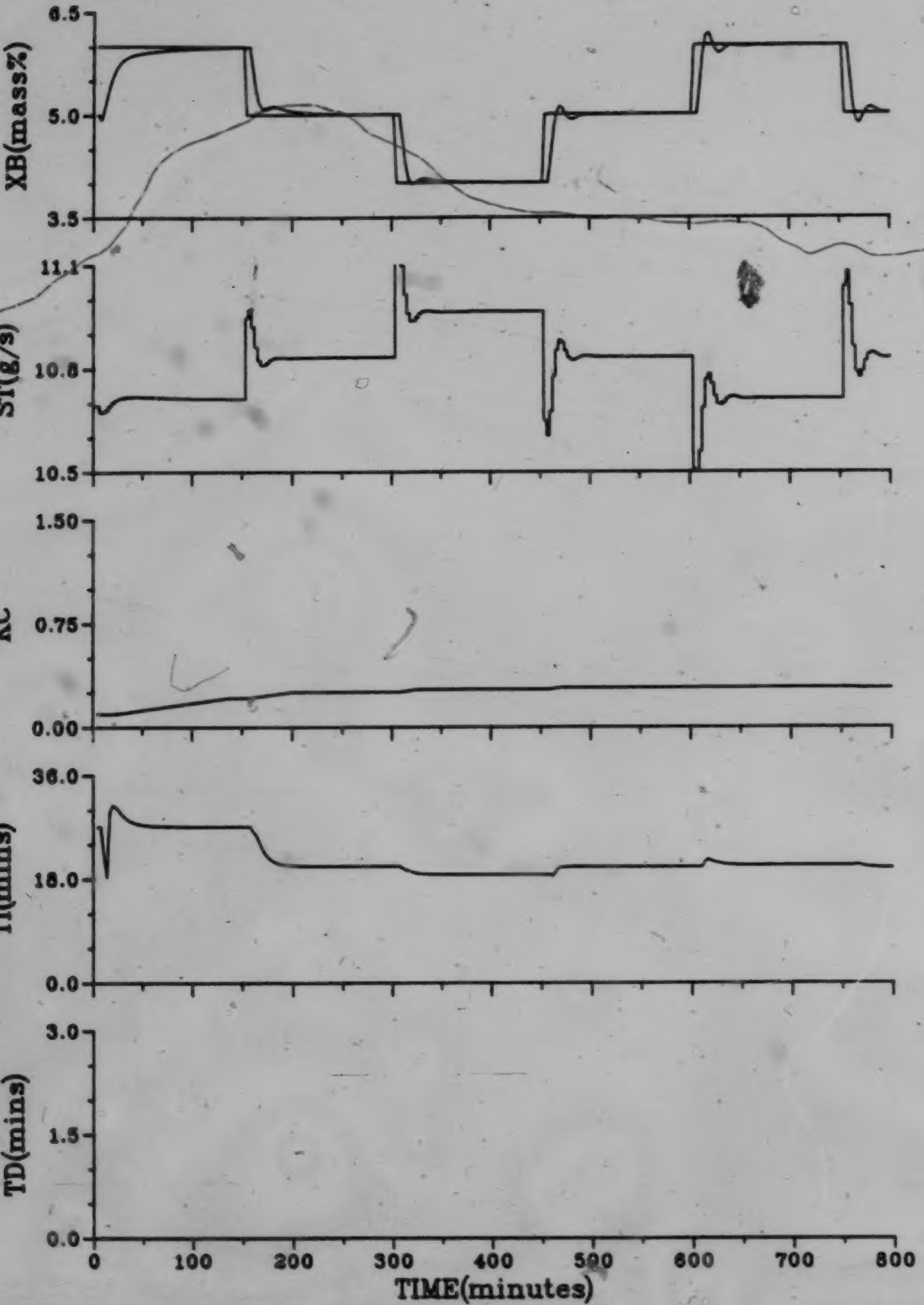

Figure 9.6a Servo self-tuning PI control of BTMS composition (INC, P=M=C=1, $M_p$sp=15, T_s=3, da=de=1, no noise)

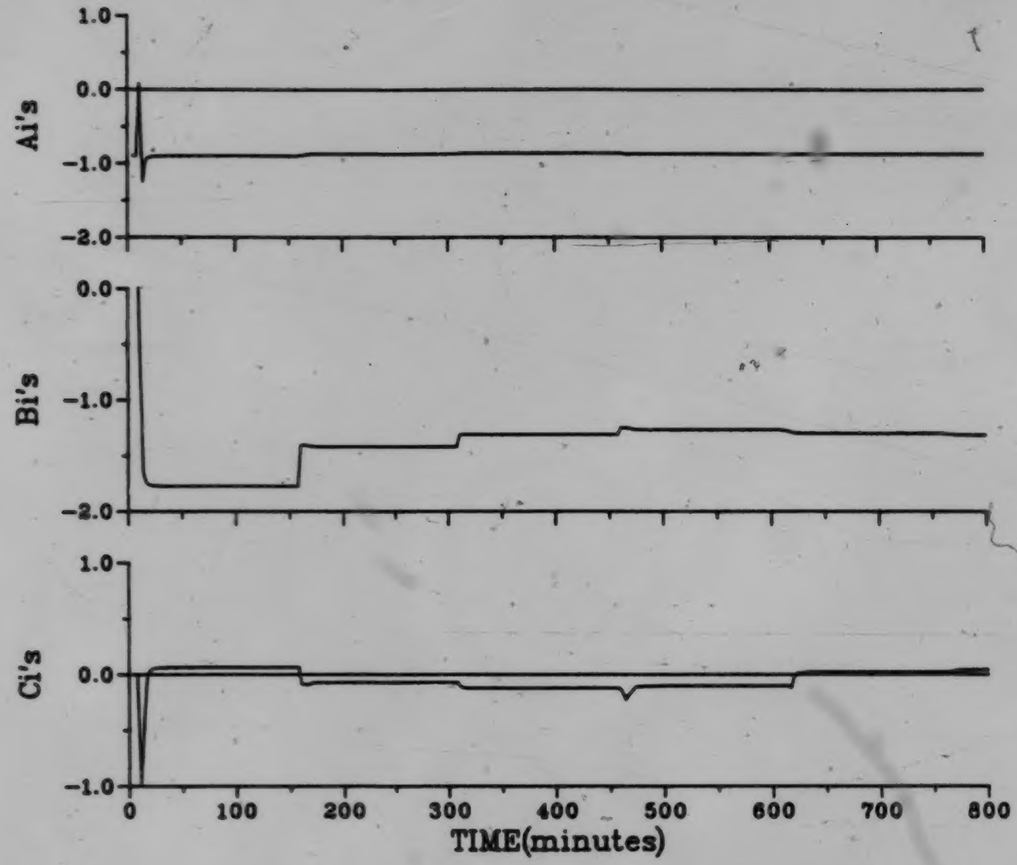

Figure 9.6b Servo self-tuning PI control of BTMS composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,da=de=1,no noise)

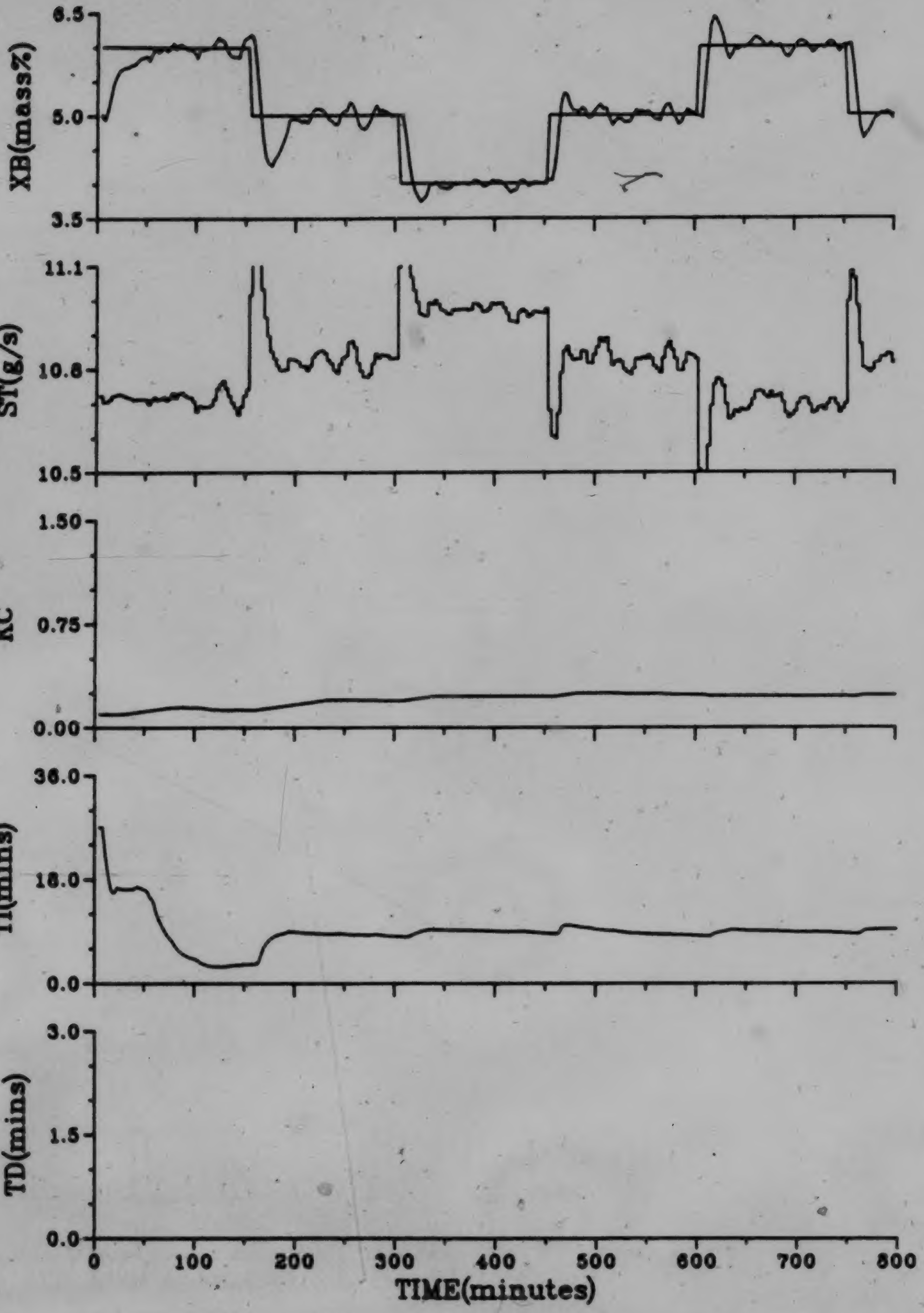

Figure 9.7a Servo self-tuning PI control of BTMS composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,da=de=1,noise)

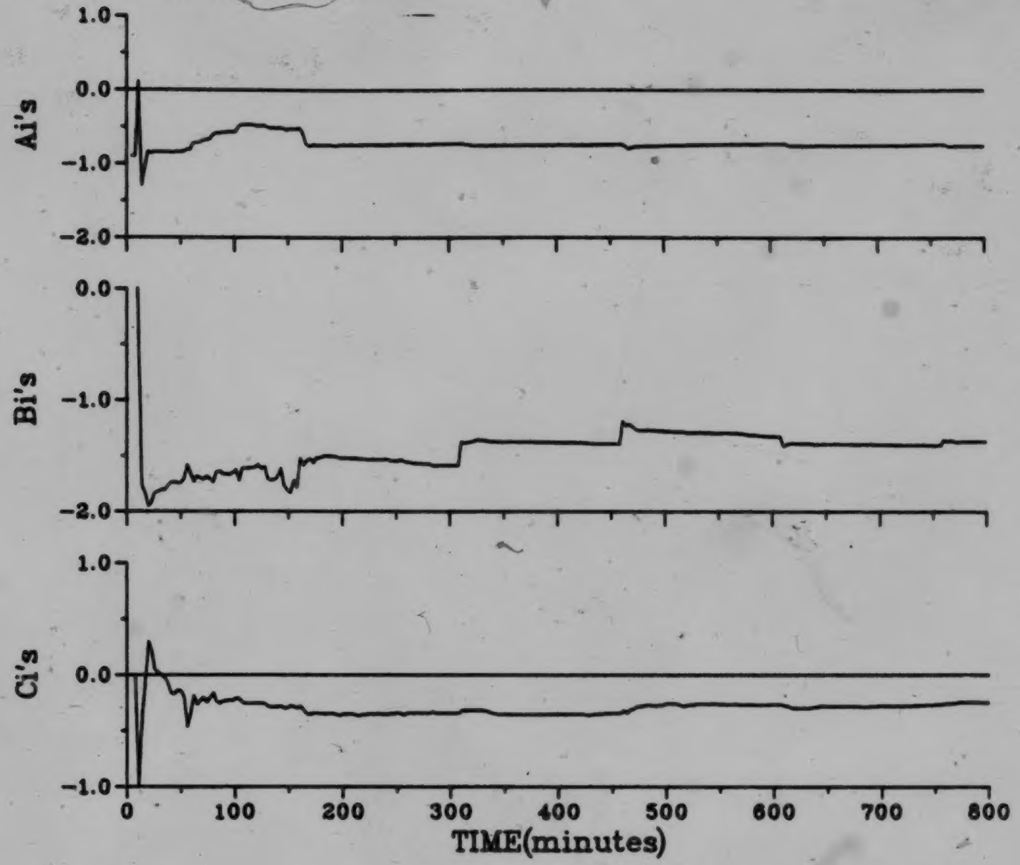

Figure 9.7b Servo self-tuning PI control of BTMS composition (INC,P=M=C=1,$M_p$sp=15,T_s=3,da=de=1,noise)

the biasing of the a_1 estimate as shown in this estimated model at T=800 minutes:

$$G(z^{-1}) = \frac{-1.371z^{-1}}{1 - 0.757z^{-1}} \tag{9.8}$$

This causes the T_i to be half the value of the no-noise run. The result on the closed-loop performance is an increase in control action and the oscillatory response seen in Figure 9.7. This again illustrates the sensitivity of incremental identification to high frequency noise.

The tuned Kc and Ti of Figure 9.7 are used in a fixed PI control algorithm to show the regulatory ability of the self-tuning PI controller tuned for setpoint changes. These results are seen in Figure 9.8. Load disturbances are a ±25% change in the liquid feed flow rate to the column. Regulatory control is good with the bottoms composition returning to within 2% of the setpoint in 10 sample intervals. Disturbance rejection for a -25% change results in an underdamped response. Comparing Figure 9.8 with Figure 9.5 it is apparent that feed flow rate changes have a much greater impact on the bottom loop than on the top loop. This is attributed to the liquid feed which has a faster effect on the stripping portion of the column.

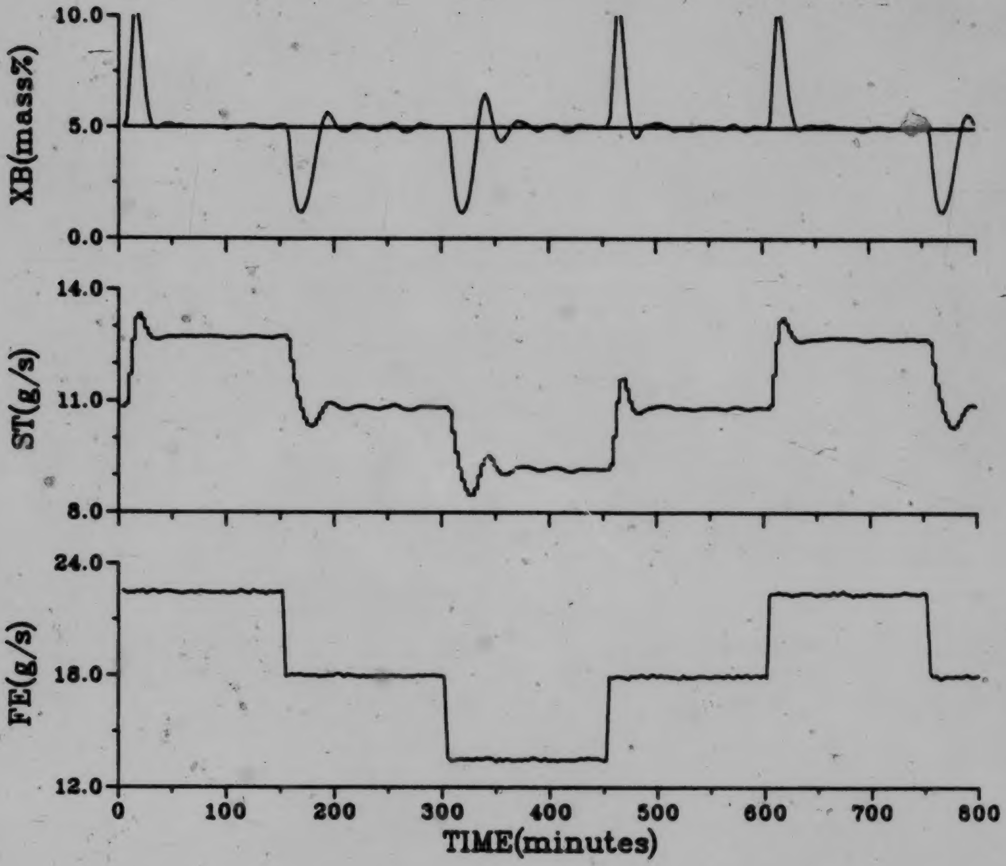

Figure 9.8 Regulatory PI control of BTMS composition (K_c=0.232,T_i=8.60,T_s=3,noise)

9.5 Summary

The distillation column simulation results show that the self-tuning PPPI controller can handle systems which approach the complexity of a real process. SISO servo and regulatory control of the overhead composition and bottoms composition show satisfactory results. It also demonstrates that the controller design strategy is feasible in self-tuning mode - the strategy which calculates the controller gain by solving the estimated characteristic equation and the integral and derivative times to cancel the estimated process poles.

The sensitivity of incremental variable identification to high frequency noise is illustrated. For simulation runs with measurement noise the parameter estimates are biased from the no noise runs. This biasing of parameter estimates must be reduced to allow long term operation of the self-tuning controller in the presence of coloured noise. One approach would be to employ a recursive estimation algorithm like recursive instrumental variables (RIV) to estimate a general noise model. All runs presented assumed a simplified $N(t)=C(z^{-1})\xi/(A(z^{-1})\Delta)$ noise model. A second procedure to minimize the noise effects is to pass the input-output data through an analog filter before passing it to the estimation algorithm. A final method is to sample the process at a slower rate.

Two other issues which became apparent during the course of these simulation studies are the importance of proper input-output data scaling and problems of overestimating model order. Initially, the process output was passed to the parameter estimation algorithm in weight fractions instead of weight percent. For the distillation colum overhead loop, this means that range through which the overhead composition varied is 0.94-0.96, a change of 0.02. The corresponding reflux flow rate variation is 9.0-11.0. Parameter estimation with this choice of units is extremely slow with a resultant poor closed-loop response. When the process output is passed to the estimation algorithm in weight percent (i.e. 94.0-96.0), parameter convergence is greatly accelerated. This phenomenon was attributed to numerical ill-conditioning of the identification algorithm (Hanson and Lawson, 1969) and underlines the importance of proper data scaling. As a general rule of thumb the range through which the input moves should be close to the range of the output signal.

The second issue encountered is the problems which can arise if the process model order is overestimated. Incremental variables identification in the presence of nonlinearities, correlated noise and an overparameterized model results in the process parameter estimates drifting to unreasonable values and in some cases unstable closed-loop responses. Incremental variables is more sensitive to this overparameterization than positional variables. If the exact process order is unknown, it is safer to underestimate the model order than to overestimate the model order.

10. Conclusions and Recommendations

10.1 Conclusions

Two levels of conclusions are presented as a result of the theoretical, stability and performance analysis done in this thesis. The first level is a small number of overall conclusions which tersely summarize the contribution and significance of this research project. The second level is made up of numerous conclusions which describe in more detail the findings of this work.

The overall conclusions are summarized below:

1. The performance of the self-tuning PI(D) controller derived from the pole placement technique is superior to that of the generalized minimum variance technique.
2. A new design method for controller gain calculation based on the closed-loop poles of the estimated characteristic polynomial is shown to be a workable scheme. It is directly related to the maximum overshoot (M_p) of the closed-loop system when used in conjunction with the pole placement design principle. For low order systems with no numerator dynamics and time delays, the closed-loop response is exactly as specified.
3. Incremental variable identification, while insensitive to process bias and deterministic step disturbances, is very sensitive to unmodelled dynamics, process nonlinearities, time delay mismatch and process

noise. Positional variable identification is sensitive to a nonzero process bias and deterministic disturbances, but shows less sensitivity to mismatch, nonlinearities and noise.

The second level of conclusions elaborate on the overall conclusions and are divided into three areas. They are summarized below:

Self-Tuning PID Controller Comparisons

1. The self-tuning PI(D) algorithm of Song et al. (1986) is rigorously derived from a ARIMA process representation. The original derivation included the *ad hoc* step of setting $C(z^{-1})=1$ whereas the ARIMA model calculates $C(z^{-1})=1$.
2. The Song et al. (1986) and the Tuffs and Clarke (1985) adaptive controllers are equivalent for the simplified PI(D) forms.
3. As observed via the root locus analysis, positioning of the controller zeroes near the process poles, as determined by T_i and T_d, accounts for the superior performance of the pole placement PI(D) controller. This draws the closed-loop pole which originates at controller integrator pole well within the unit circle.

Controller Design Considerations

The conclusions stated below are for the self-tuning PI(D) controller based on the pole placement principle and the new design method for the controller gain.

1. The effect of removing the setpoint signal from the derivative and/or proportional term is a slower servo response resulting from different closed-loop zero locations. However, the stability margin, closed-loop poles and regulatory response is identical for the three PI(D) algorithm forms.
2. In the presence of process model mismatch, the new design method for the controller gain becomes a tuning knob to shape the closed-loop response. If the unmodelled dynamics or time delay mismatch becomes large, unstable controller gains may be calculated. The design method depends on a "reasonable" model of the process.
3. The self-tuning PID controller performed well for processes with a known time delay. The controller gain is detuned to take the destabilizing effect of a time delay into account and results in a more sluggish closed-loop response.
4. The time delay compensation method of Vogel and Edgar (1980) when used in conjunction with the self-tuning PI(D) controller results in good closed-loop response for both a known and unknown time delay.
5. The presence of minimum phase numerator dynamics in the process had no adverse effect on the self-tuning controller performance.
6. The self-tuning PI(D) controller tuned for servo control gives good regulatory control for deterministic

and stochastic disturbances.

Parameter Estimation Considerations

1. Incremental and positional variable identification result in different estimated parameters in the presence of mismatch. The incremental identification model matches the initial dynamics of the process and not necessarily the steady state gain. The positional model gives a better match of the process dynamics and steady state gain and results in a better controller design.
2. The sensitivity of incremental variables identification to unmodelled dynamics and noise can be reduced by proper sampling rate selection and analog filtering of input and output signals.

10.2 Recommendations for Future Work

This section suggests topics for future research in the field of self-tuning control and specific areas where the self-tuning PI(D) controllers of this project might be improved.

1. The root locus analysis of the self-tuning PID algorithms gave valuable information about how the generalized minimum variance and pole placement principles designed their respective PID control law and how this impacted on the stability and performance characteristics of the closed-loop system. The importance of controller zeroes and pole locations

became apparent. This type of root locus analysis is recommended for adaptive controllers in general as it provides a clear picture of how the controller operates.

2. The root locus technique is only one analysis method. Other standard techniques are available for the analysis of closed-loop adaptive systems. Frequency domain techniques could be used to analyze adaptive controllers as this will provide further insight into the design method, stability and performance properties of the controller. It may also suggest further design methods for adaptive controllers. In particular, a frequency domain analysis of the self-tuning PID controllers outlining in this thesis should be completed to complement the root locus analysis.
3. Incremental variable identification is sensitive to unmodelled dynamics. This problem can be reduced by estimating a higher order model and using model reduction techniques to obtain the low order model necessary for PID controller design. In addition the proposed new design method for controller gain would give better results if more information about the process is available. Further study into the impact of model reduction techniques on self-tuning control is recommended.
4. It became apparent from the study of the various self-

tuning PID algorithms that there are problems with the controller gain calculation. This led to the proposal of a new design method for the controller gain based on the estimated closed-loop pole locations. This procedure could be refined in a number of ways. First, a more accurate method of predicting the time-domain characteristics of the closed-loop system based on the estimated location of the closed-loop poles and zeroes should be developed. Presently, an accurate prediction is only possible for up to a second order system with no numerator dynamics. Secondly, work should be done to assess the feasibility of using the actual time response data as a fine tune mode for the controller gain calculation. One possible method would be to use an expert system supervisory shell to monitor the maximum overshoot, rise time and settling time of the closed-loop system and make recommendations on how the controller gain should be modified.

5. The practical usefulness of incremental identification on real processes is limited by its sensitivity to unmodelled dynamics, nonlinearities, time delay mismatch and process noise. Studies should be completed to determine the extent this sensitivity can be reduced. Possible areas of investigation are analog filtering of input-output data and sampling rate selection. Otherwise, the utility of incremental identification is limited.

6. Use of the various recursive least squares algorithms result in biased parameter estimates in the presence of coloured noise. This affects the design of the self-tuning PI(D) controller. An investigation should be completed to assess recursive identification schemes which can estimate the parameters of a more general noise model so that unbiased parameter estimates are obtained.

References

Andreiev,N. (1981). A new dimension : a self-tuning controller that continually optimizes PID constants. *Control Engineering*, 28(8), 84-85.

Ashoor, N. and V. Singh (1982). A note on low order modelling. *IEEE Trans. Auto. Control*, AC-27, 1124-1126.

Åström,K.J. and P. Eykhoff (1971). System identification - A survey. *Automatica*, 7, 123-162.

Åström,K.J. and B.Wittenmark (1973). On self tuning control. *Automatica*, 9, 185-199.

Åström,K.J. and B.Wittenmark (1980). Self-tuning contolllers based on pole-zero placement. *IEE Proc. D*, 127(3), 120-130.

Åström,K.J. (1983). Theory and applications of adaptive control - a survey. *Automatica*, 19, 471-486.

Åström,K.J., P.Hagander and J.Sternby (1984). Zeroes of sampled systems. *Automatica*, 20, 31-38.

Åström,K.J. and T.Hägglund (1984). Automatic tuning of simple regulators for phase and amplitude margin specifications. *Automatica*, 20, 645-651.

Åström,K.J. and B.Wittenmark (1984). *Computer Controlled Systems: Theory and Design*. Englewood Cliffs, NJ: Prentice-Hall.

Åström,K.J. (1985). Auto-tuning, adaptation and expert control. *Proc. 1985 American Control Conference*, Boston, USA, 1514-1519.

Belanger,P.R. (1983). On type 1 systems and the Clarke-Gawthrop regulator. *Automatica*, 19, 91-94.

Bierman,G.J. (1976). Measurement updating using the U-D factorization. *Automatica*, 12, 375-382.

Chien,I.L., D.E.Seborg and D.A.Mellichamp (1985). A self-tuning controller for systems with unknown or varying time delays. *Int. J. Control*, 42(4), 949-964.

Clarke,D.W. and P.J.Gawthrop (1975). Self-tuning controller. *IEE Proc.*, 122(9), 929-934.

Clarke,D.W. and P.J.Gawthrop (1979). Self-tuning control. *IEE Proc.*, 126(6), 633-640.

Clarke,D.W. and P.J.Gawthrop (1981). Implementation and application of microprocessor-based self-tuners. *Automatica*, 17, 233-244.

Clarke,D.W., A.J.Hodgson and P.S.Tuffs (1983). Offset problem and k-incremental predictors in self-tuning control. *IEE Proc. D*, 130(5), 217-225.

Cluett,W.R., S.L.Shah and D.G.Fisher (1986). Verification of Rohrs' design guidelines for adaptive control. Submitted for publication to *Automatica*.

Cohen, G.H. and G.A. Coon (1953). Theoretical consideration of retarded control. *Trans. ASME*, 75, 827-834.

Davison,E.J. (1976). The robust control of a servomechanism problem for linear time-invariant multivariable systems. *IEEE Trans. Auto. Control*, AC-21, 25-34.

de la Cruz,J.M., S.Dormido and H.Ruiperez (1983). Tuning of PID controllers by model reduction. *ACI 83*, 18-6 - 18-11.

Dorf,R.C. (1980). *Modern Control Systems* Reading, MA: Addison-Wesley.

Doss, J.E. and C.F. Moore (1981). The discrete analytical predictor - A generalized dead time compensation technique. *ISA Trans.*, 20(4), 77-85.

Dumont,G.A., C.Zervos and P.R.Belanger (1985). Automatic tuning of industrial PID controllers. *Proc. 1985 American Control Conference*, Boston, USA, 1573-1578.

Fessel,J. and M.Karny (1979). Choice of models for self-tuning regulators. *Proc. IFAC Symp.on Ident. and System Para. Est.*, Darmstadt, W.Germany.

Fortescue,T.R., L.S.Kershenbaum and B.E.Ydstie (1981). Implementation of self-tuning regulators with variable forgetting factor. *Automatica*, 17, 831-835.

Francis,B.A. and W.M.Wonham (1975). The internal model principle for linear multivariable regulators. *J. Appl. Math. Opt.*, 2, 170-194.

Francis,B.A. and W.M.Wonham (1976). The internal model principle for control theory. *Automatica*, 12, 457-465.

Franklin,G.F. and J.D.Powell (1981). *Digital Control of Dynamic Systems*. Reading, MA: Addison-Wesley.

Gawthrop,P.J. (1982a). Self-tuning PI and PID controllers. *Proc. IEEE Conf. on Application of Adaptive and Multivariable Control*, Hull, England, 158-163.

Gawthrop,P.J. (1982b). A continuous-time approach to discrete-time self-tuning control. *Optimal Control Appl. and Methods*, 3(4), 399-414.

Goodwin,G.C. and K.S.Sin (1984). *Adaptive Filtering, Prediction and Control*. Englewood Cliffs, NJ: Prentice-Hall.

Gregory,P.C. (1959). *Proc. of the Self Adaptive Flight Control Systems Symp.*. WADC Tech. Rep. 59-49, Wright-Patterson Air Force Base, Ohio.

Habermayer,M. and L.Kevicsky (1985). Investigation of an adaptive Smith controller by simulation. *Proc. 1985 IFAC Conf. Digital Computer App. to Process Control*, Vienna, Austria, 413-417.

Hägglund,T (1983). The problem of forgetting old data in recursive estimation. Preprints of *IFAC Workshop on Adaptive Systems in Control and Signal Processing*, San Francisco, USA.

Hägglund,T. and K.J.Åström (1985). Automatic tuning of PID controllers based on dominant pole design. *Proc. IFAC 1985 Workshop on Adapt. Control of Chem. Processes*, Frankfurt, Germany, 212-217.

Hanson,R.J. and C.L.Lawson (1969). Extensions and applications of the householder algorithm for solving least squares problems. *Math. of Computations*, 23, 787-812.

Isermann,R. (1980). Practical aspects of process identification. *Automatica*, 16, 575-587.

Isermann,R. (1981). *Digital Control Systems*. Berlin: Springer-Verlag.

Isermann,R. (1982). Parameter adaptive control algorithms - A tutorial. *Automatica*, 18, 513-528.

Isermann,R. (1986). Parameter adaptive control systems. Submitted to *Optimal Control Systems and Applications*.

Kalman,R.E. (1958). Design of a self-optimizing control system. *Trans. ASME*, 80, 468-478.

Kan,H.W. (1982). *Binary distillation column control: Evaluation of digital control algorithms*. M.Sc. Thesis, University of Alberta, Edmonton, Alberta.

Kawashima,H. (1981). Identification of autoregressive integrating processes. *Proc. 8th IFAC World Congress*, Kyoto, Japan.

Kofahl,R. and R.Isermann (1985). A simple method for automatic tuning of PID-controllers based on process parameter estimation. *Proc. 1985 American Control Conference*, Boston, USA, 1143-1148.

Kuo,C.C, A.B.Corripio and C.L.Smith (1973). Digital control Algorithms. I. Dahlin algorithm. *Instrum. & Control Syst.*, 46(10), 57-59.

MacGregor,J.F., J.D.Wright and H.M.Hong (1975). Optimal tuning of digital PID controllers using dynamic-stochastic models. *Ind. & Eng. Chem., Process Des. & Dev.*, 14(4), 398-402.

Montague,G.A., A.J.Morris, A.R.Wright, M.Aynsley and A.Ward (1986). Modelling and adaptive control of fed-batch penicillin fermentation. *Can. J. Chem. Eng.*, 64, 567-580.

Morris,A.J., T.P.Fenton and Y.Nazer (1977). Application of self-tuning regulators to the control of chemical processes. *Proc. 5th IFAC Conf. on Digital Computer Applications to Process Control*, Hague, Netherland.

Nazer,Y. (1981). *Single and Multivariable Self-Tuning Controllers*. Ph.D. Thesis, University of Newcastle upon Tyne, Newcastle upon Tyne, England.

Nishikawa,Y., N.Sannomiya, T.Ohta and H.Tanaka (1984). A method for auto-tuning of PID control parameters. *Automatica*, 20, 321-332.

Ogata,K. (1970). *Modern Control Engineering*. Englewood Cliffs, NJ: Prentice-Hall.

Peterka,V. (1970). Adaptive digital regulation of noisy systems. *Proc. 2nd IFAC Symp. Ident. & Parameter Est.*, Prague, Paper 6.2.

Phillips,C.L. and H.T.Nagle (1984). *Digital Control System Analysis and Design*. Englewood Cliffs, NJ: Prentice-Hall.

Rohrs,C.E., M.Athans, L.Valavani and G.Stein (1984). Some design guidelines for discrete time adaptive controller. *Automatica*, 20, 653-660.

Schumann,R. and N.Christ (1979). Adaptive feedforward controllers for measurable disturbances. *Proc. 1979 JACC*, Denver, Colorado, USA, 500-506.

Schumann,R., K.H.Lachmann and R.Isermann (1981). Towards applicability of parameter adaptive control algorithms. *Proc. 8th IFAC World Congress*, Kyoto, Japan.

Seborg,D.E., S.L.Shah and T.F.Edgar (1986). Adaptive control strategies for process control : A survey. *AIChE J.*, 32, 881-913.

Seborg,D.E., T.F.Edgar and D.A.Mellichamp (1987). *Process Dynamics and Control*. Submitted for publication.

Shannon,C.E. (1949). Communication in presence of noise. *Proc. IRE*, 37, 10-21.

Silveira,H.M. and R.Doraiswami (1984). New structure for an adaptive servomechanism controller. *IEE Proc. D.*, 131(2), 64-68.

Smith, O.J.M. (1957). Closer control of loops with deadtime. *Chem. Eng. Progr.*, 53(5), 212-219.

Smith,C.A. and A.B.Corripio (1985). *Principals and Practice of Automatic Process Control*. New York: John Wiley and Sons.

Song,H.K. (1983). *Derivation and Evaluation of an Adaptive Controller*. Ph.D. Thesis, University of Alberta, Edmonton, Alberta.

Song,H.K., S.L.Shah and D.G.Fisher (1983). A self-tuning feedback controller. *Proc. 1983 American Control Conference*, San Francisco, USA, 832-837.

Song,H.K., D.G.Fisher and S.L.Shah (1984). Exerimental evaluation of a robust self-tuning PID controller. *Can.J.Chem.Eng.*, 62, 755-763.

Song,H.K., S.L.Shah and D.G.Fisher (1986). A self-tuning robust controller. *Automatica*, 22, 521-531.

Stephanopoulos,G. (1984). *Chemical Process Control - An Introduction to Theory and Practice*. Englewood Cliffs, NJ: Prentice-Hall.

Strejc,V. (1980). Least squares parameter estimation. *Automatica*, 16, 535-550.

Tjokro,S. (1984). *Derivation and Evaluation of an Adaptive PID Controller*. M.Sc. Thesis, University of Alberta, Edmonton, Alberta.

Tjokro,S. and S.L.Shah (1985). Adaptive PID control. *Proc. 1985 American Control Conference*, Boston, USA, 1528-1534.

Tuffs,P.S. (1984). Self-tuning control: algorithms and applications. D.Phil. Thesis, Oxford University, Oxford, England.

Tuffs,P.S. and D.W.Clarke (1985). Self-tuning control of offset : a unified approach. *IEE Proc. D*, 132(3), 100-110.

Vien,A. (1986). Process identification package user's manual. Internal Report, Depart. of Chem. Eng., University of Alberta, Edmonton, Alberta.

Vogel,E.F. and T.F.Edgar (1980). An adaptive deadtime compensator for process control. *Proc.1980 Joint Automatic Control Conference*, San Francisco, USA, TP5-E.

Vogel,E.F. (1982). *Adaptive Control of Chemical Processes with Variable Dead-Time*. Ph.D. Thesis, University of Texas, Austin, Texas.

Warwick,K. (1984). A new approach to reduced-order modelling. *IEE Proc. D*, 131(2), 74-78.

Warwick,K. (1985). Reduced order self-tuning control. *Proc. IFAC Conf. Identification and System Parameter Estimation*, York, UK, 1305-1309.

Wellons,M.C. and T.F.Edgar (1985). A generalized analytical predictor for process control. *Proc.1985 American Control Conference*, Boston, USA, 637-645.

Wellstead,P.E., J.M.Edmunds, D.Prager and P.Zanker (1979a). Self-tuning pole/zero assignment regulators. *Int.J.Control*, 30(1), 1-26.

Wellstead,P.E., D.Prager and P.Zanker (1979b). Pole assignment self-tuning regulator. *IEE Proc.*, 126(8), 781-786.

Wellstead,P.E. and S.P.Sanoff (1981). Extended self-tuning algorithm. *Int.J.Control*, 34(3), 433-455.

Whitaker,H.P., J.Yamron and A.Kezer (1958). Design of model reference adaptive control systems for aircraft. *Report R-164*, Instrumentation Laboratory, M.I.T., Cambridge, Massachusetts.

Wittenmark,B. (1979). Self-tuning PID controllers based on pole placement. *Report TFRT-7179*, Dept. of Auto. Control, Lund Inst. of Tech., Lund, Sweden.

Wittenmark,B. and K.J.Åström (1980). Simple self-tuning controllers. H.Unbehauen (ed.), *Methods and Applications in Adaptive Control*, Springer-Verlag, New York.

Wittenmark,B. and K.J.Åström (1984). Practical issues in the implementation of self-tuning control. *Automatica*, 20, 595-605.

Yaglom,A.M. (1973). An introduction to the theory of stationary random functions (translated by R.A.Silverman). Dover.

Ydstie,B.E., L.S.Kershenbaum and R.W.H.Sargent (1985). Theory and application of an extended horizon self-tuning controller. *AIChE J.*, 31, 1771-1780.

Young,P.C. (1972). Comment on "On-line identification of linear dynamic systems with application to Kalman filtering. *IEEE Trans. Aut. Control*, AC-17, 269-270.

Zafiriou,E. and M.Morari (1985). Digital controller for SISO systems : a review and new algorithm. *Int. J. Control*, 42(4), 855-876.

Zervos,C., P.R.Belanger and G.A.Dumont (1985). On PID controller tuning using orthonormal series identification. *Proc. 1985 IFAC Workshop on Adapt. Control of Chem. Processes*, Frankfurt, W. Germany, 13-18.

Ziegler,J.G. and N.B.Nichols (1942). Optimum settings for automatic controllers. *Trans. ASME*, 64, 759-768.

Appendix A

Table A.1 is a sample listing of the data file used to configure the nonlinear distillation column simulation package (DCSP) used in Chapter 9. Lines 2-6, 9 and 11-52 are unchanged for all of the results presented and are chosen to approximate the pilot scale binary distillation column located in the Department of Chemical Engineering.

Table A.1 Input data file for distillation column simulator (DCSP)

```
XD   SETPOINT CHANGES FOR TOP LOOP CLOSED LOOP SIMULATIONS
 30.0,                                                SAMPLE TIME
800,   10,    5,  900,    2,     1,    2,    6,
 17.980,   9.930,  10.800,  219.81,  135.59,
 3068.0, 2301.10, 0.50120, 0.95986, 0.96881,
 0.0500,  0.9500,                                     XBSP,XDSP
   1,    2,    5,  155,   305,  455,  605,  755,  905, 1055,  CHANGE TIME
 0.0, 0.0,1.053, 0.0,-1.053, 0.0,1.053, 0.0,-1.053, 0.0,  DIST
 0.9000,  0.9000,
 0.0600,  0.0900,  0.0300,  0.0300,  0.0000,
 0.04997,  0.18697,  0.35693,  0.46078,  0.49500,  XT(10)
 0.59445,  0.69810,  0.79780,  0.88306,  0.96000,
 9.1003,  22.359,  22.190,  22.998,  23.710,       LT(10)
 5.9853,  6.7409,  7.8925,  9.2761,  9.1297,
 13.259,  13.090,  13.897,  14.610,  15.115,       VT(10)
 15.871,  17.022,  18.406,  19.250,  0.0000,
 7300.0,  1001.1,  1180.0,  1101.0,  841.00,       WT(10)
 732.10,  707.20,  516.70,  507.90,  729.30,
 400.00,  406.23,  381.36,  356.49,  331.62,       QLP(10)
 306.75,  281.87,  257.00,  522.30,  0.0000,
 510.70,  533.80,  456.70,  403.00,  365.90,       QRP(10)
 332.20,  285.10,  248.00,  437.10,  0.0000,
 1.0000,  1.0000,  1.0000,  1.0000,  1.0000,       E(10)
 1.0000,  1.0000,  1.0000,  1.0000,  0.0000,
 1.0000,  0.9500,  0.9500,  0.9500,  0.9500,       EE(10)
 0.9500,  0.9500,  0.9500,  0.9500,  0.0000,
 0.0000,  0.0350,  0.0689,  0.1020,  0.1339,       XS(26)
 0.1650,  0.2026,  0.2389,  0.2739,  0.3078,
 0.3405,  0.3722,  0.4028,  0.4325,  0.4612,
 0.4892,  0.5162,  0.5665,  0.5926,  0.6400,
 0.7273,  0.8058,  0.8767,  0.9412,  0.9713,
 1.0000,
 0.0000,  0.2158,  0.3469,  0.4372,  0.5055,       YS(26)
 0.5609,  0.6143,  0.6556,  0.6870,  0.7100,
 0.7331,  0.7500,  0.7652,  0.7793,  0.7927,
 0.8043,  0.8154,  0.8300,  0.8449,  0.8624,
 0.8934,  0.9224,  0.9504,  0.9759,  0.9881,
 1.0000,
 100.00,  96.390,  93.610,  91.390,  89.050,       TLS(26)
 87.500,  85.400,  84.280,  82.700,  81.670,
 80.250,  79.610,  78.780,  78.000,  77.170,
 76.600,  75.800,  74.800,  74.060,  73.220,
 71.330,  69.440,  67.940,  66.110,  65.440,
 64.700,
 0.0000,  0.0500,  0.1000,  0.1500,  0.2000,       XST(19)
 0.2500,  0.3000,  0.3500,  0.4000,  0.4500,
 0.5000,  0.5500,  0.6000,  0.6500,  0.7000,
 0.7500,  0.8000,  0.9000,  1.0000,
 418.39,  399.56,  378.64,  355.63,  331.46,       HXS(19)
 305.43,  280.32,  256.38,  235.46,  221.75,
 203.85,  191.53,  182.93,  177.82,  173.63,
 170.61,  168.52,  165.50,  163.17,
```